

Ensembles of Many Diverse Weak Defenses can be Strong: Defending Deep Neural Networks Against Adversarial Attacks

Ying Meng, Jianhai Su, Jason O’Kane, Pooyan Jamshidi
Department of Computer Science and Engineering
University of South Carolina
Columbia, SC, USA

Abstract

Despite achieving state-of-the-art performance across many domains, machine learning systems are highly vulnerable to subtle adversarial perturbations. Although defense approaches have been proposed in recent years, many have been bypassed by even weak adversarial attacks. An early study [22] shows that ensembles created by combining multiple weak defenses (i.e., input data transformations) are still weak. We show that it is indeed possible to construct effective ensembles using weak defenses to block adversarial attacks. However, to do so requires a diverse set of such weak defenses. In this work, we propose ATHENA, an extensible framework for building effective defenses to adversarial attacks against machine learning systems. Here we conducted a comprehensive empirical study to evaluate several realizations of ATHENA. More specifically, we evaluated the effectiveness of 5 ensemble strategies with a diverse set of many weak defenses that comprise transforming the inputs (e.g., rotation, shifting, noising, denoising, and many more) before feeding them to target deep neural network (DNN) classifiers. We evaluate the effectiveness of the ensembles with adversarial examples generated by 9 various adversaries (i.e., FGSM, CW, etc.) in 4 threat models (i.e., zero-knowledge, black-box, gray-box, white-box) on MNIST. We also explain, via a comprehensive empirical study, why building defenses based on the idea of many diverse weak defenses works, when it is most effective, and what its inherent limitations and overhead are.

I. INTRODUCTION

Though they achieve state-of-the-art performance across many domains such as speech recognition [45], object detection [21], and image classification [21], machine learning systems are highly vulnerable to *adversarial examples*. Such adversarial examples, also known as *wild patterns* [4], can typically be crafted by adding small, human-imperceptible perturbations to legitimate examples [5], [17], [26], [41]. Various adversarial attacks have been recently proposed to compromise machine learning systems by leveraging such vulnerabilities [5], [11], [17], [26], [29], [32], [35]. Vulnerability of machine learning models to adversaries can lead to extremely serious security consequences. For example, self-driving vehicles may be tricked to recognize stop signs as speed limit signs [13] or a medical diagnostic device may be fooled to identify a benign tissue as malignant [16]; both scenarios may result in serious consequences [23].

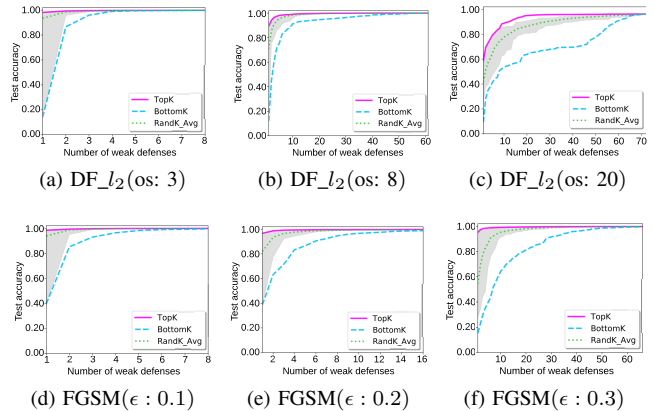


Fig. 1: Test accuracy of ensembles increase as the number of weak defenses increase and the patterns vary depending on the strength of the adversary.

In this work, we focus on evasion attacks [3], where an adversary perturbs input data at test time with the aim to mislead the machine learning classifier. A previous work has shown that ensembles of weak defenses (WD) are still weak in this type of scenario [22]. The common characteristic among all of the ensemble methods considered in [22] is the adoption of a very limited number of weak defenses (i.e., less than 3). In this work, we show that it is possible to indeed construct effective ensembles as a defense mechanism for machine learning systems if we use *many diverse* weak defenses as opposed to only a few ones. In our extensive experiments under different types of attacks, we observed that the effectiveness of an ensemble may depend on the number and diversity of WDs. We built a prototype of an ensemble with many WDs, where each WD is a DNN classifier that first applies an associated transformation¹ on the original input and then predict an output for the transformed input. We built the ATHENA prototype using an ideal ensemble strategy, wherein as long as one or more WDs output the correct label, the ensemble is able to correctly classify the input.² We trained a pool of candidate WDs associated with a large variety of transformations, evaluated each WD on all sets of adversarial examples, and then varied the number of WDs involved for the purpose of identifying and evaluating the ideal model.

To demonstrate the idea behind ATHENA, we use Figure 1

¹We therefore use *weak defense* and *transformation* interchangeably

²Note that this is only for demonstration

to present the changes in test accuracy of the ideal model with k ($k = 1; 2; 3; \dots$) weak defenses on 6 sets of adversarial examples generated by DEEPFOOL and FGSM (3 variants per attack with weak, median, and strong attacking capabilities). In the figure, we plot the results of the random selection of WDs (green dotted line) as well as best weak defenses first (pink solid line) and worst weak defenses first (blue dashed line). From Figure 1, we observe:

For adversarial examples crafted by a certain adversary, the more effective the WDs, the fewer WDs are required for the ideal model to achieve a certain level of test accuracy.

For adversarial examples generated by the same attack method (e.g., DEEPFOOL, FGSM, etc.), the stronger the adversarial example is, the more WDs are required to build an effective ideal model. For example, in order to build an ideal model to achieve a test accuracy above 95%, only two WDs are required for samples generated by DEEPFOOL(os:3) (Figure 1(a)). As the capability of the adversary increases, more WDs are required, for example, 5 and 57 WDs for DEEPFOOL(os:8) (Figure 1(b)) and DEEPFOOL(os:20) (Figure 1(c)), respectively.

Besides the quality (in terms of effectiveness against adversary) of WDs, the effectiveness of the ideal model relies on the *quantity* and *diversity* of WDs. The ideal ensemble becomes more effective as the number of WDs (k) used to build the ideal ensemble model increases, no matter how we pick and add the new weak defense to the ensemble. That is, test accuracy of an ensemble increases as it is built with a greater number of WDs. For example, evaluated with examples crafted by DEEPFOOL(os:20), the average test accuracy increases from 56.99% to 95.26%.

Such observations hold in all 9 adversaries (see Figure 13).

The observations suggest that it is possible to construct an ensemble with many diverse WDs such that, by utilizing the outputs from WDs, the model becomes robust against many forms of attacks. In this work, we aim to construct efficient defenses without presuming anything regarding the unknown inputs (i.e., the adversary used to craft the adversarial examples). That is, we are focusing on building defenses that, in general, are robust to any adversary. We, therefore, propose ATHENA³, a *flexible* and *extensible framework* for building such effective and generic defenses to adversarial attacks against machine learning systems.

To realize ATHENA, there are many potential transformations and it is impossible to involve the entire spectrum into the ensemble model. Instead, we examine various types of representative *image transformations* (e.g., rotation, shifting, noising, denoising, and many more) and implement several variants for each type of transformation; for example, we choose several rotations (90 and 180 degrees), several shifts in different directions (up, left, down, right), and so on. In total, we consider 72 transformations (complete list is in Table III).

Overall, we make the following contributions:

We studied the defensive effectiveness of 72 transformations. To the best of our knowledge, this is the first work studying such a large variety of transformations.

We propose a framework, called ATHENA, for building ensemble defenses that is flexible to scale by simply adding more WDs into or removing WDs from the ensemble. ATHENA is also very easy to update by replacing any weak defenses or the ensemble strategy being used to produce the final label from the outputs of WDs. We proposed, implemented, and evaluated 5 ensemble strategies (e.g., majority voting among WDs), showing the possibility to construct effective defenses.

We evaluated ATHENA via 4 threat models (with different assumptions about what adversary knows about the defense) for image classification with Convolutional Neural Networks (CNN) on MNIST:

- *Zero-knowledge*: We evaluated on 27 sets of adversarial examples crafted by 9 attack methods. Our proposed defense improves the effectiveness by 1.16x (83.34% to 97.09%) to 28.73x (3.43% to 98.56%) (Section IV-D).
- *Black-box*: We evaluated on 180 sets of adversarial examples (for each of the 5 ensembles with 4 various budgets and 9 attack methods). We have shown that the proposed defense can block the transfer of adversarial examples from the trained substitute model to the target classifier, where the accuracy of the target defended model significantly improves over the fooled model from 1.29x (6% to 89.12%) to 52.32x (1.9% to 99.4%) (Section IV-E).
- *Gray-box and White-box*: We generated 5 sets of adversarial examples with various constraints for each model. The effectiveness of our approach drops with adversarial examples that were generated based on gray-box/white-box threat model (test accuracy drops to 16% for the gray-box threat model and 6% for the white-box threat model). However, we show that it is computationally costly (356x for gray-box threat model and 290x for white-box threat model) and easily detectable by a simple adversarial detector (87.76% examples for gray-box and 89.8% examples for white-box threat model are successfully detected) (Section IV-F).

We performed a comprehensive empirical study to understand why the ensemble of many WDs works, when it is most effective, as well as what its overhead is.

We also released the source code and experimental data with the hope that others can build upon this extensible framework via <https://github.com/softsys4ai/athena>.

II. BACKGROUND AND DEFINITIONS

In this section, we layout notations and concepts for the remainder of the paper.

A. Notations

In this paper, we refer to the original data set as $D \in \mathbb{R}^d$; a transformation operation as T_i ($i = 1; 2; 3; \dots$); and $T_i(\mathbf{x})$ as the output of applying transformation T_i on the input \mathbf{x} . The composition of n transformations is also a transformation:

³Goddess of defense in Greek mythology

$T_g = T_n \circ T_{n-1} \circ \dots \circ T_1$. Therefore, we use T_i to denote a transformation or a composition of transformations. $D_{t_i} = \{f(\mathbf{x}) \mid \mathbf{x} \in D\}$ is used to denote the set of transformed examples of D associated to T_i . That is, $D_{t_i} = T_i(D)$.

We focus our work on *supervised machine learning*, where a classifier is trained on labeled data in D . Here, a classifier is a function $f(\cdot)$ that takes as input a data point $\mathbf{x} \in \mathcal{R}^d$ and produces a vector \mathbf{y} of probabilities associated to all classes in \mathcal{C} . Given a target model $f(\cdot)$ and an input \mathbf{x} with ground truth y_{true} , an adversary attempt to produce an *adversarial example* (AE) \mathbf{x}^0 , such that $\text{argmax}(f(\mathbf{x}^0)) \neq y_{\text{true}}$, where:

$$\begin{aligned} \mathbf{x}^0 &= \text{argmax}_{\mathbf{x} + \delta} (L(f; \mathbf{x}; y_{\text{true}})); \\ \text{whereby } \|\delta\|_p &\leq \epsilon \text{ and } \mathbf{x} + \delta \in [0; 1]^D; \end{aligned} \quad (1)$$

where $\text{argmax}(f(\mathbf{x}))$ is the predicted output given an input \mathbf{x} , L is the loss function, δ is the perturbation, and ϵ is the magnitude of the perturbation. To remain undetected, the adversarial example \mathbf{x}^0 should be as *similar* to the *benign example* as possible; therefore, attack methods (e.g., FGSM, PGD, and CW) use different norms (such as l_0 , l_2 , or l_1) to constrain the distance between \mathbf{x} and \mathbf{x}^0 .

When crafting an adversarial example for an input, some attackers force the target model to produce a specific output label, $t \in \mathcal{C}$, for a given input, while other attackers only seek to produce an output label that does not equal the ground truth. The former is referred to as a *targeted*, while the latter is referred to as an *untargeted attack* or *non-targeted attack*.

Given a set of N input data $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ and a target classifier $f(\cdot)$, an *adversarial attack* aims to generate $\{\mathbf{x}_1^0, \dots, \mathbf{x}_N^0\}$, such that each \mathbf{x}_n^0 is an adversarial example for \mathbf{x}_n . The *success rate* of an attack is measured by the proportion of predictions that were altered by an attack: $\frac{1}{N} \sum_{n=1}^N \mathbb{1}[f(\mathbf{x}_n) \neq f(\mathbf{x}_n^0)]$. The success rate is generally measured as a function of the magnitude of the perturbations performed by the attack, using the *normalized l_2 -dissimilarity*:

$$\frac{1}{N} \sum_{n=1}^N \frac{\|\mathbf{x}_n - \mathbf{x}_n^0\|_2}{\|\mathbf{x}_n\|_2}. \quad (2)$$

A strong adversarial attack has a high success rate while its normalized l_2 -dissimilarity is low.

A *defense* is a method that aims to make the prediction on an adversarial example equal to the prediction on the corresponding clean example, i.e., $\text{argmax}(f(\mathbf{x}^0)) = y_{\text{true}}$. In this work, our defense mechanism is based on the idea of *many diverse weak defenses*, which are essentially *transformation-based defenses*, i.e., they produce output label via $f(T_i(\mathbf{x}^0))$. Typically, $T(\cdot)$ is a complex, non-differentiable, and potentially stochastic function, which makes it difficult for an adversary to attack the machine learning model $f(T(\mathbf{x}))$, even when the adversary knows both $f(\cdot)$ and $T(\cdot)$.

The model f_{t_i} that was trained using data set D_{t_i} ($i = 1; 2; \dots$) is referred to as a *weak defense* (WD), and each model will be used collectively to construct defense ensembles. On the other side, the original model f that is trained on D is referred to as the *undefended model* (UM).

B. Adversarial Attack

Many adversarial attack methods have been proposed to generate “strong” adversarial examples [4].

Gradient-based Attacks perturb their input with the gradient of the loss with respect to the input. Some attacks in this family perturb the input only in one iteration. For example, FGSM [17] processes an adversarial example as following:

$$\mathbf{x}^0 = \mathbf{x} + \epsilon \text{sign}(\nabla_{\mathbf{x}} J(\mathbf{x}; \mathbf{y})); \quad (3)$$

where J is the cost function of target model f , $\nabla_{\mathbf{x}}$ is the gradient with respect to the input \mathbf{x} with corresponding true output \mathbf{y} , and ϵ is the magnitude of the perturbation.

Other variations, like BIM [27], PGD [29], and MIM [11], are iterative and gradually increase the magnitude until the input is misclassified. For example, BIM, an extension of FGSM, rather than taking one big jump ϵ , takes multiple smaller steps ϵ/α with the result clipped by α . Specifically, BIM begins with $\mathbf{x}_0^0 = \mathbf{x}$, and at each iteration it performs:

$$\mathbf{x}_i^0 = \text{clip}_{\mathbf{x}}(\mathbf{x}_{i-1}^0 + \epsilon \text{sign}(\nabla_{\mathbf{x}} J(\mathbf{x}_{i-1}^0; \mathbf{y}))); \quad (4)$$

where $\text{clip}_{\mathbf{x}}(\mathbf{A})$ denotes the element-wise clipping of \mathbf{x} ; the range of \mathbf{A}_{i-1} after clipping will be $[\mathbf{x} - \alpha; \mathbf{x} + \alpha]$.

JSMA [35], another gradient-based approach, greedily finds the most sensitive direction, such that changing its values will significantly increase the likelihood of a target model labeling the input as the target class:

$$\begin{aligned} S_t &= \frac{t}{x_i}; S_o = \sum_{j \neq t} \frac{j}{x_j}; \\ s(x_i) &= S_t / S_o^j \quad (S_t < 0) \quad (S_o > 0); \end{aligned} \quad (5)$$

where S_t represents the Jacobian of target class $t \in \mathcal{C}$ with respect to the input image, and where S_o is the sum of Jacobian values of all non-target classes.

Optimization-based Attacks generate adversarial examples by solving an optimization problem like:

$$\text{argmin}(d(\mathbf{s}; \mathbf{x} + \delta) + c \cdot L(\mathbf{x} + \delta)); \quad (6)$$

where L is the loss function for solving $f(\mathbf{x} + \delta) = t$ and $t \in \mathcal{C}$ is the target label. CW [5] constraint adversarial samples to stay within a certain distance from the benign example. DEEPFOOL [32] takes iterative steps to the direction of the gradient provided by a locally linear approximation of the target model.

Black-box Attacks do not require any knowledge of target models, yet are shown to be effective in fooling machine learning models. ONE-PIXEL [40], one of the extreme adversarial attack methods, generates adversarial examples using an evolutionary algorithm called Differential Evolution [39], which fits well with gradient-free optimization. ONE-PIXEL represents individuals as a vector of real numbers, new individuals are generated as follows:

$$x_c = x_{p1} + \text{mutation}(x_{p2}, x_{p3}); \quad (7)$$

where x_c is a child and x_{pi} ($i = 1; 2; 3$) are parents in the evolutionary algorithm.

(a) Training weak defenses

(b) Producing output for a given input

Fig. 2: ATHENA: A framework of Many Weak Defenses.

III. ATHENA: A FRAMEWORK OF MANY WEAK DEFENSES

In this section, we present our defense framework called ATHENA, based on which we can construct effective defenses that are resilient to adversarial attacks, even with full knowledge of the defense in place. Further, such defenses are not tied to a particular machine learning (ML) classification and can be deployed in different domains outside of image classification. The defense is based on ensemble of many diverse weak defenses. To be upfront, we will evaluate a particular realization of ATHENA based on 72 WDs and 5 ensemble strategies for image classification with CNN models.

A. Approach Overview

ATHENA works as follow: At test time, for the given input data x , (i) it first collects outputs from all WDs, and then (ii) it uses an ensemble strategy (e.g., majority voting) to compute the final output (Figure 2(b)). Each WD (orange dashed rectangles in Figure 2) can be any ML model (such as CNN or Logistic Regression) trained separately (Figure 2(a)).

Our proposed defense framework is:

- Extensible: One can add new WDs and, therefore, improve its effectiveness or remove WDs, which will sacrifice the effectiveness but will also decrease the runtime overhead.
- Flexible: One can update the ensemble, making it robust by replacing any (i) WDs and/or (ii) ensemble strategy.

In the section, we will describe the ingredients of our framework, including the essence of WDs (Section III-B) as well as the ensemble strategies (Section III-C).

B. Transformation as a Weak Defense

Transformations have shown to be effective in deterring some adversarial perturbations [2], [20], [38]. To provide more comprehensive understanding of the effectiveness of

Fig. 3: Sample adversarial images generated by various adversaries and their corresponding distortions created by transformations represented in each row. (i) In the 1st row, we present inputs—an original input followed by the adversarial examples with predicted label and confidence produced by the UM (). (ii) In the 2nd row, we present the perturbation generated by the adversary corresponding to the column. (iii) In each cell in rows 3–9, we present the distortion generated by applying the transformation represented in a current row to the input in the column with the predicted label and confidence produced by the weak defense associated to the corresponding transformation.

transformations, we examined a large variety of transformations (see Table III) against AEs generated by various attacks. Figure 3 presents sample inputs from MNIST in the 1st row—a legitimate input followed by AEs generated by the delineated adversaries that correspond to each column, the corresponding adversarial perturbations (δ_k , $x - x^0$) are shown in the 2nd row, and the distortions (δ_k , $T_i(x^0)$) generated by some sample transformations are shown in the 3rd–9th rows. Under each sample, the predicted label and the corresponding probability produced by the WD when fed with the input in the 1st row are plotted. For example, the sample in grid (5, 5) was generated by applying Iter(max) on the AE that had been generated by DPFOOL_{l2} and the corresponding WD labeled it as the digit 7 with a confidence of 0.28. Overall, the following observations have been made:

- For an AE, there is at least one WD that can correctly label it. For example, the AE generated by DPFOOL is correctly classified by all WDs except geometric(iradon).
- Each WD is able to correctly classify some of the AEs generated by an adversary. For example, WD associated to morphology(dilation) can correctly classify samples generated by FGSM, BIM_{l1}, CW_{l2}, JSMA, MIM, and PGD.
- Different transformations are effective against different type of adversaries. For example, the sample generated by FGSM is misclassified by WD associated to rotation(0); however, it is correctly classified by morphology(dilation).

These observations alongside the earlier observations summarized in Figure 1 indicate that ensembles on top of many diverse transformations could form effective defenses.

C. Ensemble of Many Diverse Weak Defenses

As observed in Figure 3, the effectiveness of a certain transformation varies according to the adversary being used to craft the AEs. It is difficult to find a small set of transformation operations (i.e., 2–3) that are effective against all adversaries. Nevertheless, the observations from Figures 1 and 13 suggest that the weakness of individual WDs can be addressed by increasing the quantity of sample WDs included in the ensemble.

As presented in Figure 2(a), we train a WD by first applying a transformation T_i on the original data D , and then training a CNN classifier using the transformed data set D_{T_i} . At the testing phase (Figure 2(b)), given an input x (it can be a legitimate input or an adversarial input), we first apply a transformation T_i on the input x and then feed the transformed input x_{T_i} to corresponding WD D_{T_i} . After we collected predictions from all WDs, we use an ensemble strategy (e.g., majority voting) to compute the final predicted label by utilizing the predictions (probabilities and/or logits) of WDs. We will elaborate on 5 ensemble strategies in Section IV-B.

IV. EXPERIMENTAL EVALUATION

Here, we describe the experimental setup and discuss the results pertaining to the effectiveness of AENA.

A. Experimental Setup

TABLE I: Architecture and training parameters of weak defenses and undefended model.

Architecture			Parameters		
Conv.ReLU	3	3	32	Optimization Method	Adam
Max Pooling	2	2		Learning Rate	0.001
Conv.ReLU	3	3	64	Batch Size	128
Max Pooling	2	2		Epochs	50
Dense	4096				
Dropout	0.4				
Dense	10				
Softmax	10				

- 1) Dataset We evaluated our work thoroughly on MNIST, a set of handwritten digits, containing 28 × 28 gray-scale images in the training and testing set. We used this benchmark due to its extensive use by adversarial ML work, whereby a comprehensive evaluation was feasible with this benchmark.
- 2) Attack methods: To evaluate our approach, we crafted 3 sets of AEs with different densities of perturbations (weak, medium, strong) for each of the 9 attack methods (created 27 AE sets in total): FGSM, BIM (norm and l_1 -norm versions), JSMA, PGD, DEFPOOL (l_2 -norm), CW (l_2 -norm), ONE-PIXEL, and MIM. We implemented attack methods on top of CleverHans [33].
- 3) We trained a list of WDs, each associated to one transformation (Table III). When training a classifier, we used 80% of training data as our training examples and 20% as the validation examples. In this work, we evaluated our approach only on CNNs (LeNet architecture). The details about the architecture and the training parameters are in Table I. We used TensorFlow for training and inference.
- 4) We used 5 ensemble strategies (Section IV-B).

5) We evaluated the effectiveness of our defense in four forms of threat models from the weakest to the strongest: zero-knowledge (Section IV-D), black-box (Section IV-E), gray-box (Section IV-F), and white-box (Section IV-F).

6) We conducted the experiments (e.g., training WDs, crafting AEs) on multiple machines (Table IV).

B. Ensemble Strategies

We implemented 5 simple WD ensemble strategies:

- 1) Random Defense (RD) strategy will randomly choose a WD to predict the input. The expected effectiveness of this ensemble is the average of all WDs. Where most of the WDs chosen are robust, the resulting ensemble using RD strategy is also expected to be robust. This strategy has an additional benefit of stochastic behavior in which neither the attacker nor the defender know what WD will be used at a certain time; so, the attacker has to fool all WDs.
- 2) Majority Voting (MV) strategy will collect the predicted labels of all WDs, and then it determines the label that is agreed upon by most WDs. The more evenly the predicted labels are distributed, the fewer WDs are required for the ensemble to correctly classify the input. MV requires only $\frac{\max_i f_{T_i, g_i}}{2} + 1$ WDs to correctly predict the correct label.
- 3) Top 2 Majority Voting (T2MV) strategy works similar to the MV except that T2MV collects labels associated to the top two probabilities and, thereafter, performs majority voting among them. As we observed from Figure 3, the predicted probability distributions of some WDs on certain AEs are soft, such that those WDs are less confident with their predictions, whereby the correct answer may lie in the classes for which they are less confident.
- 4) Average Probability (AVEP) strategy produces the final predictions based on the average predicted probabilities of all WDs for a given input. Although some WDs are not confident with their outputs for some AEs, other WDs are able to correctly classify some AEs with high confidence. AVEP takes advantages of the expertise of the latter WDs in order to find a label with the highest average confidence. AVEP collects predicted probabilities of all WDs and then take an average of the outputs; finally, AVEP returns the predicted label with the highest probability value.
- 5) Average Logits (AVEL) is another strategy based on average outputs of all WDs. Some information may get lost through use of the sigmoid transformation in CNNs [24]. Therefore, AVEL utilizes the information hidden in the logits to produce the output label with the highest average logits; the calculation of the output is similar to AVEP.

C. Threat Models

In this work, we make the following assumptions:

- 1) The attackers can attack only at the inference/testing phase. The attackers can modify only the input data; neither the models nor the training data can be modified by attackers.
- 2) Depending on adversary's knowledge of the ensemble (architectures and hyper-parameters of WDs, ensemble strategies, etc.), we considered four different threat models [4], [6]:

Fig. 4: Test accuracy of individual weak defenses and ensembles against various adversarial test data sets.

are likely recovered by a greater number of transformations. For example, with small normalized dissimilarities (cf., Figure 10), AEs that were generated by CW, BIM_{I2}, BIM_{I1}, MIM, and PGD attacks as well as AEs generated by FGSM, DEFPOOL, and ONE-PIXEL with small perturbed magnitudes are recovered by most WDs. As the distance between AEs and BS increases, less WDs are able to recover the AEs.

E. Evaluation against Black-box Attacks

Under the black-box threat model, the attackers' knowledge regarding the system is only based on information gained by querying the target classifier. Thus attackers construct a substitute model f_{sub} , mimicking the original target model f_{ens} using only a limited training data; they use the substitute model to create AEs and to launch attacks on the system. This strong assumption challenges the attacker in many ways when crafting an AE, such as selecting a technique for building the substitute model and preparing a sample set for label collection given a limited query budget [19]. Although we could consider different assumptions for the knowledge of the attacker in black-box attacks, we instead considered the strongest possible black-box attack to make our evaluation even stronger, where the adversary knows the exact DNN architecture, optimization strategy, and hyper-parameters of the original target model. The attacker can then use the substitute model's parameters and perform a white-box attack on the substitute model. Finally, we evaluate the accuracy of the original model, meaning that we test whether these AEs transfer to the original model. More specifically, as shown in Figure 5, we evaluate the black-box threat model as follows:

- 1) Collect a data set of N samples, $D_{bb} = \{(x; y) | y = f_{ens}(x); x \in D_g\}$, by querying the ensemble model t times.
- 2) Build a substitute model f_{sub} , mimicking the original target model f_{ens} which is trained on the collected data D_{bb} .

Fig. 5: The evaluation process of a black-box threat model.

For each ensemble model we trained a substitute model, each of which uses a set of random samples from a subset of the MNIST test set and their corresponding labels that have been collected from the target ensemble model. The MNIST subset had k samples, and from this set, we created 4 sets containing 50, 100, 500, and samples with uniform distribution across classes (we created even more extreme cases with a very low budget 10 or a higher budget and we presented the results in Figure 20). The remaining samples in MNIST test set were used for the evaluation of the defense in this threat model.

Once we trained the substitute model associated to each budget, we used the parameters of the trained model to launch attacks against the substitute model. For each substitute model, we created AEs using the 9 adversarial attacks as listed in Table II, and we used the generated AEs to attack the target ensemble model. Therefore, for a target ensemble model, 36 sets of AEs (9 attack types each having 4 different budgets) were created for evaluation. To limit the computational cost, we used 1k samples out of 10k test samples for generating AEs. We used the accuracy improvement of the target ensemble model over a substitute model when they are under the same

⁴It may be argued suspicious if the attacker query the system many times

Algorithm 1: Crafting an adversarial example in white-box/gray-box threat model

```

input : x, y, attacker, N, max_dissimilarity
1 F_fooled ← EMPTY SET;
2 F_cand ← all weak defenses;
3 x0 ← x;
4 while size(F_fooled) < N do
5   f_target ← pickTarget(F_cand, strategy);
6   perturbation ← attacker.getPerturbation(x0);
7   xtmp ← x0 + perturbation
8   if dissimilarity(xtmp; x) > max_dissimilarity then
9     break;
10  end
11  for fti in F_cand do
12    if y ≠ fti(xtmp) then
13      addModel(F_fooled, fti);
14      removeModel(F_cand, fti);
15    end
16  end
17  x0 ← xtmp;
18 end
return x0;

```

Fig. 6: Model accuracy when under attack in black-box. The red square on the FGS axis in the bottom-right radar chart indicates that the MV ensemble holds roughly 70% accuracy under the attacking scenario where the attacker has a query budget of 1000.

attack to indicate the effectiveness of our proposed defense. Here, we report the detailed results of targeting the RD ensemble and the MV ensemble. The results of the other ensembles we tested are similar, and, therefore, we provide these additional results in the Appendix (see Figure 21).

Overall, for the RD ensemble, the accuracy improvement falls within the range [20.12%; 92.83%] with a 57% average. The accuracy of the RD ensemble is calculated over 100 runs. The accuracy distribution of the 100 runs for each attack and each budget are shown in Figures 22 and 23. For the other four ensembles, the accuracy improvement falls in the range [26.3%; 97.5%] with a 65.28% average. Based on results in Figure 6, we make the following observations:

Even though the generated attack was successful in undermining the substitute models, most generated AEs were not able to transfer to the defense system; both the RD and MV ensembles were able to sustain a relatively high accuracy under the generated attacks.

- More specifically, when targeting both the RD and MV ensembles, each type of AEs (except One-Pixel) was able to effectively attack substitute models, thereby limiting their accuracy below 35%. But the RD ensemble sustains its test accuracy above 50% for 31 out of 36 AE sets and above 70% for 20 out of 36 sets, while the MV ensemble is able to maintain its accuracy above 50% for 33 out of 36 sets and 70% 28 out of 36 AE sets.

The accuracy of the target ensemble decreases when the query budget increases (except with JSMA). With a higher budget, the attacker is able to build a more accurate surrogate model and, therefore, craft more effective AEs, which could make the defense suffer from a lower accuracy.

The observations show that ATHENA could effectively fight against the black-box adversaries. It is important to emphasize that the results could have been even better for the defense mechanism under a weaker black-box threat model, where the knowledge of DNN architecture and hyper-

parameters is not known by the adversary, such that the attacker confronts greater difficulty in crafting effective AEs.

F. Evaluation against Gray-box and White-box Attacks

Here we use a greedy approach to evaluate the effectiveness of ATHENA under gray-box and white-box threat models. Using the procedure in Algorithm 1, we generated a set of AEs;

each of the AEs is able to fool (i) all weak defenses for a gray-box model or (ii) a certain number, N , of weak defenses for a white-box model within a specific distance (i.e., normalized dissimilarity between perturbed and clean input as computed by Formula 2). The value of N depends on the ensemble strategy being used and the attack approach. For example, for an ensemble model using a MV strategy, the attacker must fool at least 90% of the WDs if they generate AEs using an untargeted attack. The use of a maximum dissimilarity ensures the sample is not being perturbed too much. The maximum dissimilarity is also a factor that determines the efforts that the attacker can afford for generating AEs.

Algorithm 1 presents the core procedure for crafting a single AE x^0 for a given clean input x by fooling at most N WDs within a specific distance from the clean input (i.e., \max_{d_2} —the maximum normalized dissimilarity between x^0 and x). The set of WDs being fooled, F_{fooled} , by the current perturbed sample x^{tmp} , is initially empty. The set of WDs yet to be fooled, F_{cand} , is initially all the WDs used to build the defense ensemble. The adversarial example starts from x and is perturbed iteratively until at least N WDs have been fooled or the dissimilarity between x^{tmp} and x is greater than the constraint of \max_{d_2} . At each iteration, the target model can be selected via different strategies. For example, the attacker can use (i) a random candidate WD, (ii) the most effective candidate WD against the attack method, or (iii) the least effective candidate WD against the attack method as their target model to perturb in the current iteration. In the algorithm, any attack method (e.g., FGSM, EDP, LDF, etc.) can be used to craft adversarial perturbations. The function $\text{dissimilarity}(x^0; x)$ returns the normalized d_2

TABLE II: The attack methods used for crafting adversarial examples and their parameters under black-box threat model.

Attack Type	BIM_I ₂	BIM_I ₁	CW_I ₂	DF_I ₂	FGSM	JSMA	MIM	One-Pixel	PGD
Crafting Configuration	= 2	= 0:3	LearningRate = 1:0	Overshoot = 0	= 0:3	= 0:5; = 0:7	= 0:3	PixelCount = 30	= 0:3

(a) Evaluation of Gray-box (b) Evaluation of White-box

Fig. 7: Evaluation results for (a) Gray-box and (b) White-box threat models by detector, MV ensemble, and a simple supreme model combining the detector and the MV ensemble (cf., detection + MV ens): (i) percentage of detected adversarial examples by detector (blue lines), (ii) test accuracy of the MV ensemble (green lines), and (iii) as compared to the maximum normalized dissimilarity of AE sets.

dissimilarity (defined in Formula 2) between two samples (i.e., $N = 1$), and the function $\text{getPerturbation}(x)$ returns the generated perturbation, $\delta x = x \otimes \mathbf{Q}_2$.

Here are some specific settings we used for evaluation:

We evaluate an ensemble built on 72 WDs using only the MV ensemble strategy.

In order to add perturbations related to a variety of transformations into a single AE within a reasonable computational time, we use FGSM ($= 0:1$). Using smaller ϵ means it will take much longer to generate an AE that is able to fool just one WD. In our experiments, by adding perturbations iteratively that are guaranteed to fool one WD, it takes 4 iterations on average to craft a single AE.

In a gray-box model, with no knowledge of the ensemble strategy, the attacker aims to generate AEs that are able to fool all WDs (i.e., 72 WDs), such that the AEs are likely to fail any strategy, as none of the WDs work correctly. In a white-box model, however, the attacker knows that the MV strategy is used and wants to perform an untargeted attack; therefore, the attacker needs to fool at least 90% (72 \times 90% = 65) of the WDs to ensure the MV ensemble cannot recover the correct label.

To generate AEs with similar adversarial strength to FGSM AEs based on UM (see Figure 10), we set the distance constraint—max_dissimilarity—for the 10 AE sets from 0:1 to 1:0 with an interval of 0:1.

We randomly pick a target model from $\mathcal{M}_{\text{cand}}$

Based on the results in Figure 7 (as well as Figures 17 and 18), we make the following observations:

With modest effort, the attacker is able to generate less costly AEs. It takes on average less than 35 seconds to generate a single AE for white-box model (Figure 17) and less than 50 seconds for gray-box model (Figure 18). However, such AEs are not strong enough to fool the MV ensemble. As presented in Figure 7, the MV ensemble achieves a test accuracy exceeding 90% and is robust against such AEs,

which are too weak to fool many WDs. For example, on

average, each AE with max_dissimilarity = 0:1 cannot even fool a single WD, even with max_dissimilarity = 0:4; where as each AE on average can only fool 10% (7=72) of WDs, although 90% of WDs can recover the correct labels. On the other spectrum, as the constraint of max_dissimilarity relaxes, the attacker is able to generate stronger AEs. The test accuracy of MV ensemble drops from 99:44% (Figure 4) to 16% against gray-box and 6% against white-box attack. The MV ensemble model is ineffective as (i) most of the WDs are fooled by the AEs and (ii) AEs are very dissimilar to BS. As presented in Figures 17 and 18, on average, a single AE is able to fool 63:43 out of 72 (i.e., 88:09%) WDs for gray-box and 62:74 (i.e., 87:14%) WDs for white-box. Detailed information about the distribution of these statistics can be found in Figures 17 and 18. When most of the WDs produce incorrect outputs, our ensemble model is likely to be fooled. However, this comes at a price:

- High cost Although attackers are able to generate AEs that fool the ensemble, it is extremely expensive for them to achieve this goal. For example, in order to craft a single AE, it takes 747:77 seconds for the gray-box threat model and 610:35 seconds for the white-box threat model.
- Easily detectable Furthermore, although such AEs successfully fool the MV ensemble, they are perturbed heavily⁵ and very likely to be detected either by a human (see Figure 16) or a detector. We used an existing detector [15] to test our hypothesis regarding the possibility of detecting such AEs with high confidence. As shown in Figure 7, the detector is able to successfully detect 87:76% of AEs for the gray-box and 89:8% of AEs for the white-box. An enhanced version of our framework with the detector (Detection + MV ensemble) can achieve a high accuracy on AEs with any attacking strength by either detecting and/or recovering the correct label via WDs. The combination of detector and MV ensemble achieves a test accuracy above 90% for all AEs (except when max_dissimilarity = 0:7) for both threat models.

V. RESEARCH QUESTIONS

As shown in our extensive evaluations, STAE is effective against a wide variety of adversaries from the weakest threat model (zero-knowledge) to the strongest one (white-box). Here we investigate why and how a transformation is able to block an adversary? Does one or a set of transformations exist that are generally effective against any adversary? Does the effectiveness of an ensemble rely on the number and diversity

⁵The average normalized dissimilarity of AEs is 0:883 for gray-box and 0:851 for white-box) is larger than that of the strongest FGSM (0:864).

of WDs? What are the costs of our ensemble defense? In particular, we wish to address the following research questions:

- 1) RQ1: Can transformations work as a weak defense?
 - a) How does a transformation or composition block the effectiveness of an adversarial perturbation?
 - b) Does the robustness of a weak defense depend on the type of adversarial attack?
- 2) RQ2: Can we construct an effective ensemble using many diverse weak defenses?
 - a) Will an ensemble's effectiveness rely on the number of weak defenses?
 - b) Will ensemble's effectiveness rely on the ensemble strategy it uses?
- 3) RQ3: What is the overhead of our proposed framework?

Fig. 8: The impact of 3 WDs on CW_{L2} AEs originated from class 9. AEs are represented in hollow squares while BS are represented solid circles.

VI. RQ 1. EFFECTIVENESS OF WDs

A. How Does a Transformation Block the Effectiveness of an Adversarial Perturbation?

1) Study Design: AEs are meaningful, yet imperceptible perturbed features of the data distribution that induce the classifier to make erroneous predictions [25]; in order to understand how a transformation augments the features such that it helps to recover the “distorted” features for the classifier to make the correct prediction decision, we used t-SNE [43]. t-SNE is a technique that uses joint probability distributions to describe the closeness of data points and to visualize the classification results of a data set. In our experiment, the dataset D contains 50 AEs (generated with the same attack), which originated from 50 BS of a target class (e.g., class 9) as well as 250 BS from the target class. In addition, it further contains 300 BS for each of the other classes. We investigated the impact of our framework on 9 types of AEs and in 10 classes in MNIST. For each type of AE, we first ranked the 72 WDs in terms of classification accuracy on the same type of AEs crafted from the MNIST testing set (i.e., 10k samples), and then picked three WDs—one from the top three, one from the medium three, and one from the the bottom three. The representation of a sample fed into t-SNE is the probability prediction of the sample by either the UM or a WD.

Fig. 9: Rank correlation between the accuracy of WDs in different AEs.

2) Results: Here, we report only the result of investigating the impact of three WDs on the CW_{L2} AEs, originated from the BS of class 9. The three WDs are the ones associated with transformations, Iter minimum (bottom performer), noise speckle (medium performer) and ip both (top performer). Figure 8 (a) illustrates the classification result by the undefended model, while the other three sub-plots show the classification results by the corresponding WDs. Overall, we emphasize the following observations from our results:

class 3, while the bottom performer, WD (Iter minimum), could successfully recover these misclassified AEs back to the original class 9. A similar case can be observed by comparing WDs (noise speckle and ip both). These two interesting findings are also confirmed in the result of other AE types (see Figure 24 in the Appendix).

Summary. Transformations are able to block an adversarial perturbation by recovering the “distorted” features, such that the AE could fall closer to the group of BSs bearing its original label.

B. Why Diversity of WDs Matters?

1) Study Design: To answer this question, we investigate whether WDs perform similarly in two different adversarial attack scenarios. In more specific terms, we calculate the rank correlation between a list of the respective performances of the WDs in the context of different adversarial attacks. A positive rank correlation indicates that the listed WDs perform similarly, while a negative rank correlation indicates that they behaved differently, and a small value of rank correlation means that the effectiveness of the listed WDs has little relation between the two types of attacks. We conducted the experiment with 72 WDs and 27 sets of AEs that were

the same as these AEs' original labels. Comparing Figure 8 (a) with Figure 8 (b), (c), or (d), it can be observed that the majority of the AEs are classified as class 9 (square 9), which they have successfully recovered. A set of WDs could complement each other. The top performer, WD (ip both), still misclassifies some AEs into

Fig. 10: Normalized ℓ_2 dissimilarity between BS and each AE type.

(a) BS

(b) CW (learning rate: 0.015)

generated in a zero-knowledge threat model. We computed the Spearman's rank correlation between classification accuracy of the 72 WDs for each pair of different AE sets.

2) Results: The Spearman's rank correlation coefficients are plotted in Figure 9, where the size of a square is determined based on the absolute value of the correlation coefficient. For two sets of AEs within the same group of attack (except for the pair, FGSM ($\epsilon = 0:1$) and FGSM ($\epsilon = 0:3$)), the correlation of the WDs' performances tends to be strong, which could be observed from the nine 3×3 sub-matrices along the diagonal in Figure 9. In the exceptional pair, the difference between the two sets of AEs is the perturbation level—a fair amount of this difference could result in a significant difference in dissimilarity when compared with the BS. As seen in Figure 10, the dissimilarity of FGSM ($\epsilon = 0:3$) is 2.5x higher than that of FGSM ($\epsilon = 0:1$). Combining Figures 9 and 10, it also could be observed that if their dissimilarity values are close, this set of WDs perform similarly (high correlation) between two different sets of AEs under the same attack group, e.g., the group of BIM₂ AEs. Therefore, we believe that the dissimilarity of AEs plays an important role

But is the dissimilarity the only factor that impacts the performance similarity of WDs? The answer is no. As seen in Figure 9, the difference of dissimilarity values between BIM₂ ($\epsilon = 0:75$) and BIM₂ ($\epsilon = 1:0$) is larger than the difference between BIM₂ ($\epsilon = 0:75$) and CW₂ (learning rate = 0:01), while the performance of the WDs in BIM₂ ($\epsilon = 0:75$) is much more related to BIM₂ ($\epsilon = 1:0$) compared to that of CW₂ (learning rate = 0:01). This indicates that the intrinsic similarity of two sets of AEs (which resulted from different types of adversarial attacks) is a critical factor and indeed more important than the dissimilarity value. This conclusion is affirmed in Figure 9 by the strong correlations between the performances of WDs in the listed AE types, FGSM ($\epsilon = 0:1$), BIM₁, MIM, and PGD. As a matter of fact, FGSM is a simplified version of BIM₁, MIM, and PGD, and they each belong to the same family of gradient-based attacks. Another important observation is that, for AE sets belong to different attack groups, the WDs tend to behave differently (namely, either low or negative correlation.)

The observations above drive us to reach the conclusion that it could be sufficient to select a small group of WDs when defending against a group of AEs that come from the same kind of attacks and have close dissimilarity values since

(c) DeepFool (overshoot: 20)

(d) FGSM ($\epsilon = 0:3$)

Fig. 11: Test accuracy ensembles built with various numbers of WDs.

the selected WDs could defend against each type of AEs with a similar level of perturbation. However, such a defense type will be ineffective in practice, because we usually have no knowledge of the incoming AEs, let alone do we operate from the strict assumption, made by this defense as to the type and dissimilarity values of incoming AEs.

Summary. A diverse set of weak defenses is indeed necessary to build a robust defense against adversarial attacks.

VII. RQ 2. EFFECTIVENESS OF ENSEMBLES

A. Does the Effectiveness of an Ensemble Defense Rely on the Number of WDs?

1) Study Design: To address this question, we start with one random WD and then build a new ensemble by adding a new random WD into the current ensemble. We repeated this step until all 72 WDs had been used for building the ensemble. After the results of all 72 ensembles had been collected, we evaluated them with the 27 AE sets that were generated in a zero-knowledge threat model. We performed this process 5 times and computed the average accuracy.

2) Results: Figure 11 (refer to Figure 14 for the complete results) presents the average accuracy against each AE (the strongest among each adversary). We observed that—

(a) In general, with the exception of the RD strategy, the larger amount of WDs, the more robust the ensemble became against all attacks.

(b) For the RD strategy, building the ensemble with a greater number of WDs did not improve its robustness when considering that its expected effectiveness is the average effectiveness of all WDs. However, this ensemble strategy has the benefit of randomness, which makes it difficult for the attacker to identify what selection of WDs will be used at test time.

(c) The effectiveness of T2MV converges much slower than other strategies which result is even more pronounced

especially when most WDs are effective. One reason for this result is that the T2MV strategy requires more WDs as a conduction for providing enough correct information to overcome the distraction from noise. BS, for example, requires at least 30 WDs in order for the T2MV ensemble to achieve a test accuracy comparable to ensembles using other strategies. When using other strategies, the defense ensemble only requires less than 5 WDs to converge.

- 4) When all ensemble strategies were evaluated with AEs generated by DEEFOOL(os:20), the effectiveness of ensembles first increases when we add more WDs. At a certain point though, if we keep adding new WDs, its effectiveness starts decreasing, and it further converges at a lower test accuracy. As observed in the column labeled as "DEEFOOL(os:20)" in Figure 4, the effectiveness of most WDs drop significantly compared to other adversaries—65.28% achieve a test accuracy lower than 40%.

Summary. Subject to the exception of ensembles that use the RD strategy, in general, the greater number of WDs being employed in a defense ensemble, the more robust it will be.

B. Is There Any Ensemble Strategy that Always Outperforms Other Ensembles?

- 1) Study Design: In order to address questions regarding an ensemble's effectiveness and ensemble strategies, we built ensembles of various sizes for each ensemble strategy and then evaluated the ensembles against BS as well as sets of AEs (in the same manner as was done for research question VII-A).
- 2) Results: As presented in Figure 4 and Figure 11, we make the following observations:

No single strategy is always the most effective against all attacks. But for most cases, the RD strategy is the least effective strategy after the performance of all defenses converge. The specialized knowledge of individual WDs and their respective logits values can help improve the robustness of ensembles in the context of AEs that are heavily perturbed. However, these types of information become noise when the ensemble is evaluated with AEs that are less perturbed. For example, when evaluated with the top 2 dissimilar AE sets of FGSM($\epsilon:0.3$) (0:8364) and DEEFOOL(os:20) (0:5875)—AVEP and AVEL are much more effective than other strategies. In contrast, when evaluated with CW(lr:0.015), MV and T2MV strategies are more effective than AVEP and AVEL for the AE set that is closest to BS (1216).

Summary. No single ensemble strategy is the most robust against all adversaries. However, AVEL, with the addition of increased specialized information, works well in all cases and achieves either the highest accuracy or an accuracy comparable to the other evaluated strategies.

VIII. RQ 3. OVERHEAD OF ENSEMBLE

- 1) Study Design: Here, we specifically look into two types of overhead: memory overhead and online time overhead. The main cause of memory overhead results from loading the

Fig. 12: Overhead of transformations, inference, and ensemble evaluation.

neural network model associated to each WD. It is clear that memory consumption will be roughly N times of that used by the UM if N WDs are adopted. For online time overhead, the UM just requires the cost of inference, while AENA consist of transformations, inference calculations for each WD, and neural calculations for determining the output label.

We evaluated, in a GCP machine (see Table IV), the online time overhead of the 5 ensembles, which are based on a set of 72 WDs and 9 types of AEs by comparing the inference time of the UM with the time costs of the three aforementioned components.

2) Results: As seen in Figure 12 (b), the time cost of each of the 5 ensembles is trivial when compared to the inference time of the UM. For instance, the inference time of the UM is around 10 times of the time cost of the majority-voting based ensemble. Because the set of selected WDs could be deployed in parallel, the online time overhead in the set of WDs is dominated by the slowest WD insofar, as it consumes the largest amount of time for transformation and inference. Figure 12 (b) also illustrates that the inference time of the transformation model in each WD is larger than the inference time of the UM but at the same magnitude level. For example, the largest inference time of the transformation model for FGSM($\epsilon:0.3$) is about 1.6x the inference time of the UM. Since the time cost of inference time in WDs and ensemble strategy is on par with the inference time of the UM, the online time overhead will be impacted most by the slowest transformation. As seen in Figure 12 (a), a large and diverse set of 50 out of the 72 WDs is available for ensemble design with a time cost similar to the inference time of the UM. Another key point is that the design set of WDs can be adjusted to trade off the ensemble performance with online time overhead, which tradeoff may be necessary for time-critical applications.

Summary. The memory overhead AENA is high and is proportional to the number of WDs used in the ensemble; however, the inference time in the framework is on par with the inference time of the original undefended model.

IX. DISCUSSIONS, LIMITATIONS AND FUTURE WORK

Even though our defense increases model robustness, the following several areas are known as areas for improvement: Flexible defense framework. While our defense causes a significant increase in model accuracy in a variety of settings,

there are cases where the accuracy of the model is still low. The robustness of machine learning systems depends not only on the model and the dominant supervised learning paradigm [25], but also on the characteristics of data, such as the dimensionality of the input data [14]. As a result, existing defenses against adversarial attacks are often limited to a specific model or model-agnostic. Several defenses exist that increase the robustness of models through the supervised training process. For example, several forms of adversarial training defense increase the amount of perturbation needed to achieve a certain level of accuracy, it may even further serve as an aid for detection based defenses.

We constructed and evaluated approaches either detect adversarial examples before feeding them to classifiers [18], [31]; attempt to remove adversarial perturbations from the input [2], [9], [12], [20], [28] or feature transformations depending on the strength of the defenses [44], [46]; or combining both detection and removal [30]. Model-specific defenses make strong assumptions (e.g., about the norm they use for creating adversarial examples) about the type and nature of the adversary; therefore, the adversary can alter its attack to circumvent such model-specific defenses. Model-agnostic defenses have also been shown insufficient for removing adversarial perturbations from input data [7]. Even small ensembles of model-agnostic defenses have been shown to be brittle even against non-adaptive adversaries [22]. In this work, we provide for the first time, a strong guarantee that the composition of many and diverse sets of weak defenses is indeed effective to block adversarial attacks. The proposed framework has the capability to unify and incorporate existing and even future weak defenses into the ensemble model and make the defense mechanism even stronger.

Since we decided to evaluate the performance of our framework and empirically understand why such a defense works, empirically in depth, we decided to perform all experiments on MNIST data set. Even though we have some evidence that the observations still hold in larger data sets such as CIFAR10 and ImageNet, we plan to perform such evaluation, in depth, as a future work. We will consider different model architectures, classifiers not necessarily restricted to DNNs, while also considering different domains (medical imaging and voice), and possibly more variety of attacks targeting transformations, e.g., [1]. Investigating how ATHENA may enable automated testing of machine learning systems is considered as a fruitful future direction [36].

We plan to explore these directions in future work.

X. RELATED WORK

There has been extensive research on developing defenses against adversarial attacks; however, they have mainly focused on specific model families or application domains. In addition, the current defenses provide improved robustness against only known attacks, where the adversary possesses no or limited knowledge about the details of the defense, such that it is unclear if these defense mechanisms will be effective against adversaries with complete knowledge of their existences.

XI. CONCLUSIONS

We proposed and empirically investigated the ensemble of many diverse weak defenses as a defense against evasion attacks. We evaluated the proposed defense against state-of-the-art adversarial attacks under zero-knowledge, black-box, gray-box and white-box threat models and found out that our defense makes the target model more robust against evasion attacks. To the best of our knowledge, even though there has been previous work on using transformation as a defense, our empirical study is the first contribution that provides an understanding of using many diverse transformations to make machine learning systems more robust against evasion attacks and has some viable potential properties: (1) applicability across multiple machine learning models, (2) applicability in different domains (image, voice, video), and (3) agnosticism to particular attacks. Further, the tunability of our defense allows a system designer to pick a suitable number and type of WDs as well as a proper ensemble strategy that accounts for the appropriate utility-security trade-off based on the application.

XII. ACKNOWLEDGEMENT

This research is partially supported by Google (via GCP cloud research grant) and NASA (via EPSCoR 521340-SC001). We would like to thank Blake Edwards, Stephen Raione, Rabina Phuyel for their contributions in the defense pipeline and Marilyn Gartley for proofreading the paper.

REFERENCES

- [1] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *arXiv:1707.07397* 2017.
- [2] Arjun Nitin Bhagoji, Daniel Cullina, Chawin Sitawarin, and Prateek Mittal. Enhancing robustness of machine learning systems via data transformations. In 2018 52nd Annual Conference on Information Sciences and Systems (CISS), pages 1–5. IEEE, 2018.
- [3] Battista Biggio, Iginio Corona, Davide Maiorca, Blaine Nelson, Nedim Srndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. *Joint European conference on machine learning and knowledge discovery in databases*, 2013.
- [4] Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition* 84:317–331, 2018.
- [5] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (S&P)* 2017.
- [6] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, and Aleksander Madry. On evaluating adversarial robustness. *arXiv:1902.06705* 2019.
- [7] Nicholas Carlini and David Wagner. Magnet and “efficient defenses against adversarial attacks” are not robust to adversarial examples. *arXiv:1711.08478* 2017.
- [8] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 854–863. JMLR. org, 2017.
- [9] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Li Chen, Michael E Kounavis, and Duen Horng Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv:1705.02900* 2017.
- [10] Ambra Demontis, Marco Melis, Maura Pintor, Matthew Jagielski, Battista Biggio, Alina Oprea, Cristina Nita-Rotaru, and Fabio Roli. Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In *28th USENIX Security Symposium* 2019.
- [11] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Xiaolin Hu, and Junjie Zhu. Discovering adversarial examples with momentum. *CoRR abs/1710.06081*, 2017.
- [12] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. A study of the effect of jpg compression on adversarial images. *arXiv:1608.00853* 2016.
- [13] Kevin Eykholt, Ivan Evtimov, Earlene Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) June 2018*.
- [14] Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Analysis of classifiers’ robustness to adversarial perturbations. *Machine Learning* 107(3):481–508, 2018.
- [15] Reuben Feinman, Ryan Curtin, Saurabh Shintre, and Andrew Gardner. Detecting adversarial samples from artifacts. *arXiv:1703.00410* 2017.
- [16] Samuel G Finlayson, John D Bowers, Joichi Ito, Jonathan L Zittrain, Andrew L Beam, and Isaac S Kohane. Adversarial attacks on medical machine learning. *Science* 363(6433):1287–1289, 2019.
- [17] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv:1412.6572* 2014.
- [18] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv:1702.06280* 2017.
- [19] Chuan Guo, Jacob R Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Q Weinberger. Simple black-box adversarial attacks. *arXiv:1905.07121* 2019.
- [20] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv:1711.00117* 2017.
- [21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) June 2016*.
- [22] Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defense: Ensembles of weak defenses are not strong. In *11th USENIX Workshop on Offensive Technologies* 2017.
- [23] Douglas Heaven. Why deep-learning aIs are so easy to nature. *Nature* 574(7777):163, 2019.
- [24] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv:1503.02531* 2015.
- [25] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *arXiv:1905.02175* 2019.
- [26] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv:1611.01236* 2016.
- [27] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CLR Workshop 2017*.
- [28] Jiajun Lu, Hussein Sibai, Evan Fabry, and David Forsyth. No need to worry about adversarial examples in object detection in autonomous vehicles. *arXiv:1707.03501* 2017.
- [29] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations* 2018.
- [30] Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* 2017.
- [31] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv:1702.04267* 2017.
- [32] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. *CoRR abs/1511.04599*, 2015.
- [33] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv:1610.00768* 2018.
- [34] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv:1605.07277* 2016.
- [35] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016.
- [36] Kexin Pei, Shiqi Wang, Yuchi Tian, Justin Whitehouse, Carl Vondrick, Yinzhi Cao, Baishakhi Ray, Suman Jana, and Junfeng Yang. Bringing engineering rigor to deep learning. *ACM SIGOPS Operating Systems Review* 53(1):59–67, 2019.
- [37] Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing* 307:195–204, 2018.
- [38] Ying Cai Shixin Tian, Guolei Yang. Detecting adversarial examples through image transformation. *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [39] Rainer Storn and Kenneth Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization* 11, 1997.
- [40] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *CoRR abs/1710.08864*, 2017.
- [41] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv:1312.6199* 2013.
- [42] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv:1805.12152* 2018.
- [43] L.J.P. van der Maaten and G.E. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research* 9(Nov):2579–2605, 2008.
- [44] Qinglong Wang, Wenbo Guo, Kaixuan Zhang, Xinyu Xing, C Lee Giles, and Xue Liu. Random feature nullification for adversary resistant deep architecture. *arXiv:1610.01239* 2016.
- [45] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Achieving human parity in conversational speech recognition. *CoRR abs/1610.05256*, 2016.
- [46] Fei Zhang, Patrick PK Chan, Battista Biggio, Daniel S Yeung, and Fabio Roli. Adversarial feature selection against evasion attacks. *IEEE transactions on cybernetics* 46(3):766–777, 2015.

APPENDIX

TABLE III: Transformation operations

Category	Description
Rotation	Rotate an input by a certain angle: rotate 90 ; 180 ; and 270 .
Flip	Flip an input horizontally and/or vertically.
Shift	Shift an input in a direction by some pixels: shift left, right, up, down, top-left, top-right, bottom-left, and bottom-right.
Cartoonify	Cartoonify applies a sequence of image process operations on the input, including bilateral filter, gray scaling, median blur, create edge mask, and add the mask to original input.
Affine	Transform an input by mapping variables (e.g., pixel intensity values at position $(x_1; y_1)$) into new variables (e.g., $(x_2; y_2)$) by applying a linear combination of shift, rotation, scaling and/or shearing operations: compress vertically and/or horizontally, stretch vertically and/or horizontally.
Denoise	Noise is generally considered to be a random variable with zero mean. Denoise transformations average out noises from inputs: nl_means_fast, nl_means, tv_menas, tv_chambolle, and wavelet.
Morphology	Morphological transformations apply operations based on image shape: erosion, dilation, opening, closing, and gradient.
Noise	Add noises to an input: gaussian, localvar, pepper, poison, salt, and salt&peper.
Augmentation	Real-time data augmentation: feature-wise std normalization and sample-wise std normalization.
Segmentation	Segmentation divides the image into groups of pixels based on specific criteria. We segment an input based on colors.
Quantization	Quantization reduces the number of colors in an input using k-mean technique: 4 clusters and 8 clusters.
Distortion	Distortion deforms the pixel grid in an input and maps the deformed grid to the destination image: distort by x-axis or y-axis.
Filter	Filter transformations smooth an input using various filter kernels: entropy, gaussian, maximum, median, minimum, prewitt, rank, scharr, roberts, sobel.
Compress	Save images in different image formats: jpeg (quality: 80%, 50%, 30%, and 10%), png(compression: 1, 5, and 8).
Geometric	Apply different geometric transformations to images: iradon, iradon_sart, and swirl

(a) Benign Samples

(b) BIM_l2(: 1:2)

(c) BIM_l1 (: 0:12)

(d) CW_l2(learning rate:0.01)5

(e) DEEPFOOL_l2(os:20)

(f) FGSM(: 0:3)

(g) JSMA(: 0:21)

(h) ONE-PIXEL(pixel count: 30)

(i) MIM(: 0:1)

(j) PGD(: 0:1)

Fig. 13: Test accuracy of ensembles increase as the number of weak defenses being used to construct an ensemble increases. Such pattern exists in all types of attacks.

TABLE IV: Hardware configurations for our experiments.

Experiment	Platform
Training WDs	PC 12 Intel(R) Core(TM)i7-8700 CPU @ 3.20GHz, 32 GB memory
Crafting AEs zero-knowledge & black-box	Google Cloud VM: 8 Intel(R)Xeon(R) CPU @ 2.20GHz, 30 GB memory, 1 x NVIDIA Tesla P100 16 Intel(R)Xeon(R) CPU @ 2.30GHz, 14.4 GB memory
gray-box & white-box	PC 12 Intel(R) Core(TM)i7-8700 CPU @ 3.20GHz, 32 GB memory
Examining overheads	Google Cloud VM: 16 Intel(R)Xeon(R) CPU @ 2.30GHz, 14.4 GB memory

TABLE V: Test accuracies of substitute models.

Query Budget	10	50	100	500	1,000	5,000
RD Ensemble	40.09%	78.63%	83.20%	94.24%	95.91%	98.06%
MV Ensemble	39.43%	76.84%	85.28%	94.47%	95.62%	98.24%
T2MV Ensemble	36.51%	76.11%	83.77%	93.48%	95.40%	98.44%
AVEP Ensemble	40.39%	73.38%	82.96%	94.13%	95.59%	98.12%
AVEL Ensemble	39.36%	77.22%	83.80%	94.11%	95.52%	98.42%

(a) Benign Samples (b) BIM_L2(: 1:2)

Fig. 16: Sample adversarial examples generated in gray-box and white-box threat models with max_dissimilarity of 0:1-1:0.

(c) BIM_L1 (: 0:12) (d) CW (learning rate: 0.015)

(a) Time cost (seconds) (b) Normalized_l2 dissimilarity

(e) DEEPFOOL (overshoot: 20) (f) FGSM(: 0:3)

(c) Number of iterations (Algorithm 1)(d) Number of WDs being fooled

(g) JSMA(: 0:21) (h) ONE-PIXEL (pixel count:30)

Fig. 17: Cost evaluation of AEs generated based on white-box model (MV strategy).

(i) MIM (: 0:1) (j) PGD(: 0:1)

Fig. 14: Test accuracy ensembles built with various numbers of weak defenses.

(a) Time cost (seconds) (b) Normalized_l2 dissimilarity

Fig. 15: Sample adversarial examples generated by various adversaries.

(c) Number of iterations (Algorithm 1)(d) Number of WDs being fooled

Fig. 18: Cost evaluation of AEs generated based on gray-box model (MV strategy).

Fig. 19: Test accuracy of transformation composition of 3 lters.

Fig. 21: Model accuracy when under attack in black-box threat model.

Fig. 22: The distribution of RD ensemble's accuracy over 100 runs per attack.

Fig. 20: Accuracy of surrogate model and target ensemble when under attack. The surrogate models are built with query budgets of 10 and 5000.

Fig. 23: The distribution of RD ensemble's accuracy over 100 runs per budget.

