ResNets Ensemble via the Feynman-Kac Formalism to Improve Natural and Robust Accuracies

Bao Wang
Department of Mathematics
University of California, Los Angeles
wangbaonj@gmail.com

Bingjie Yuan
Computer Science Department
Tsinghua University
ybj14@mails.tsinghua.edu.cn

Zuoqiang Shi
Yau Mathematical Sciences Center
Tsinghua University
zqshi@tsinghua.edu.cn

Stanley J. Osher
Department of Mathematics
University of California, Los Angeles
sjo@math.ucla.edu

June 28, 2019

Abstract

Empirical adversarial risk minimization (EARM) is a widely used mathematical framework to robustly train deep neural nets (DNNs) that are resistant to adversarial attacks. However, both natural and robust accuracies, in classifying clean and adversarial images, respectively, of the trained robust models are far from satisfactory. In this work, we unify the theory of optimal control of transport equations with the practice of training and testing of ResNets. Based on this unified viewpoint, we propose a simple yet effective ResNets ensemble algorithm to boost the accuracy of the robustly trained model on both clean and adversarial images. The proposed algorithm consists of two components: First, we modify the base ResNets by injecting a variance specified Gaussian noise to the output of each residual mapping. Second, we average over the production of multiple jointly trained modified ResNets to get the final prediction. These two steps give an approximation to the Feynman-Kac formula for representing the solution of a transport equation with viscosity, or a convection-diffusion equation. For the CIFAR10 benchmark, this simple algorithm leads to a robust model with a natural accuracy of **85.62**% on clean images and a robust accuracy of **57.94**% under the 20 iterations of the IFGSM attack, which outperforms the current state-of-the-art in defending against IFGSM attack on the CIFAR10. Both natural and robust accuracies of the proposed ResNets ensemble can be improved dynamically as the building block ResNet advances. The code is available at: `https://github.com/BaoWangMath/EnResNet`.

1 Introduction

Deep learning (DL) achieves great success in image and speech perception [32]. Residual learning revolutionizes the deep neural nets (DNNs) architecture design and makes training of the ultra-deep, up to more than one thousand layers, DNNs practical [21]. The idea of residual learning motivates the development of a good number of related powerful DNNs, e.g., Pre-activated ResNet [22], ResNeXt [56], DenseNet [23], and many others. Neural nets ensemble is a learning paradigm where many DNNs are jointly used to improve the performance of individual DNNs [20].

Despite the extraordinary success of DNNs in image and speech recognition, their vulnerability to adversarial attacks raises concerns when applying them to security-critical tasks, e.g., autonomous cars [3, 1], robotics [18], and DNN-based malware detection systems [42, 17]. Since the seminal work of Szegedy et al. [51], recent research shows that DNNs are vulnerable to many kinds of adversarial attacks including physical, poisoning, and inference attacks [11, 9, 41, 16, 24, 6, 5]. The physical attacks occur during the data acquisition, the poisoning and inference attacks happen during the training and testing phases of machine learning (ML), respectively.

The adversarial attacks have been successful in both white-box and black-box scenarios. In white-box attacks, the adversarial attacks have access to the architecture and weights of the DNNs. In black-box attacks, the attacks have no access to the details of the underlying model. Black-box attacks are successful because one can perturb an image to cause its misclassification on one DNN, and the same perturbed image also has a significant chance to be misclassified by another DNN; this is known as transferability of adversarial examples [43]. Due to this transferability, it is straightforward to attack DNNs in a black-box fashion [36, 7]. There exist universal perturbations that can imperceptibly perturb any image and cause misclassification for any given network [39]. Dou et al. [13], analyzed the efficiency of many adversarial attacks for a large variety of DNNs. Recently, there has been much work on defending against these universal perturbations [4].

The empirical adversarial risk minimization (EARM) is one of the most successful mathematical frameworks for certified adversarial defense. Under the EARM framework, adversarial defense for ℓ_∞ norm based inference attacks can be formulated as solving the following EARM [38, 57]

$$\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \max_{\|\mathbf{x}_i'-\mathbf{x}_i\|_\infty\leq\epsilon} L(f(\mathbf{x}_i', \mathbf{w}), y_i), \tag{1}$$

where $f(\cdot, \mathbf{w})$ is a function in the hypothesis class $\mathcal{H}$, e.g., ResNets, parameterized by $\mathbf{w}$. Here, $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ are n i.i.d. data-label pairs drawn from some high dimensional unknown distribution $\mathcal{D}$, $L(f(\mathbf{x}_i, \mathbf{w}), y_i)$ is the loss associated with f on the data-label pair $(\mathbf{x}_i, y_i)$. For classification, L is typically selected to be the cross-entropy loss; for regression, the root mean square error is commonly used. The adversarial defense for other measure based attacks can be formulated similarly. As a comparison, empirical risk minimization (ERM) is used to train models in a natural fashion that generalize well on the clean data, where ERM is to solve the following optimization problem

$$\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} L(f(\mathbf{x}_i, \mathbf{w}), y_i). \tag{2}$$

Many of the existing works try to defend against the inference attacks by finding a good approximation to the loss function in EARM. Project gradient descent (PGD) adversarial training is a representative work along this side that approximate EARM by replacing $\mathbf{x}_i'$ with the adversarial data that obtained by applying the PGD attack to the clean data [16, 38, 40]. Zhang et al. [59] replace the empirical adversarial risk by a linear combination of empirical and empirical adversarial risks. Besides finding a good surrogate to approximate the empirical adversarial risk, under the EARM framework, we can also improve the hypothesis class to improve the adversarial robustness of the trained robust models.

1.1 Our Contribution

The robustly trained DNNs usually more resistant to adversarial attacks, however, they are much less accurate on clean images than the naturally trained models. A natural question is

Can we improve both natural and robust accuracies of the robustly trained DNNs?

In this work, we unify the training and testing of ResNets with the theory of transport equations (TEs). This unified viewpoint enables us to interpret the adversarial vulnerability of ResNets as the irregularity, which will be defined later, of the TE's solution. Based on this observation, we propose a new ResNets ensemble algorithm based on the Feynman-Kac formula. In a nutshell, the proposed algorithm consists of two essential components. First, for each $l = 1, 2, \cdots, M$ with M being the number of residual mappings in the ResNet, we modify the l-th residual mapping from $\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l)$ (Fig. 1 (a)) to $\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l) + N(0, \sigma^2\mathbf{I})$ (Fig. 1 (b)), where $\mathbf{x}_l$ is the input, $\mathcal{F}$ is the residual mapping and $N(0, \sigma^2\mathbf{I})$ is Gaussian noise with a specially designed variance σ^2. Second, we average over multiple jointly and robustly trained modified ResNets' outputs to get the final prediction (Fig. 2). This ensemble algorithm improves the base model's accuracy on both clean and adversarial data. The advantages of the proposed algorithm are summarized as follows:

- It outperforms the current state-of-the-art in defending against inference attacks.

- It improves the natural accuracy of the adversarially trained models.
- Its defense capability can be improved dynamically as the base ResNet advances.
- It enables to train and integrate an ultra-large DNN for adversarial defense with a limited GPU memory.
- It is motivated from partial differential equation (PDE) theory, which introduces a new way to defend against adversarial attacks, and it is a complement to many other existing adversarial defenses.

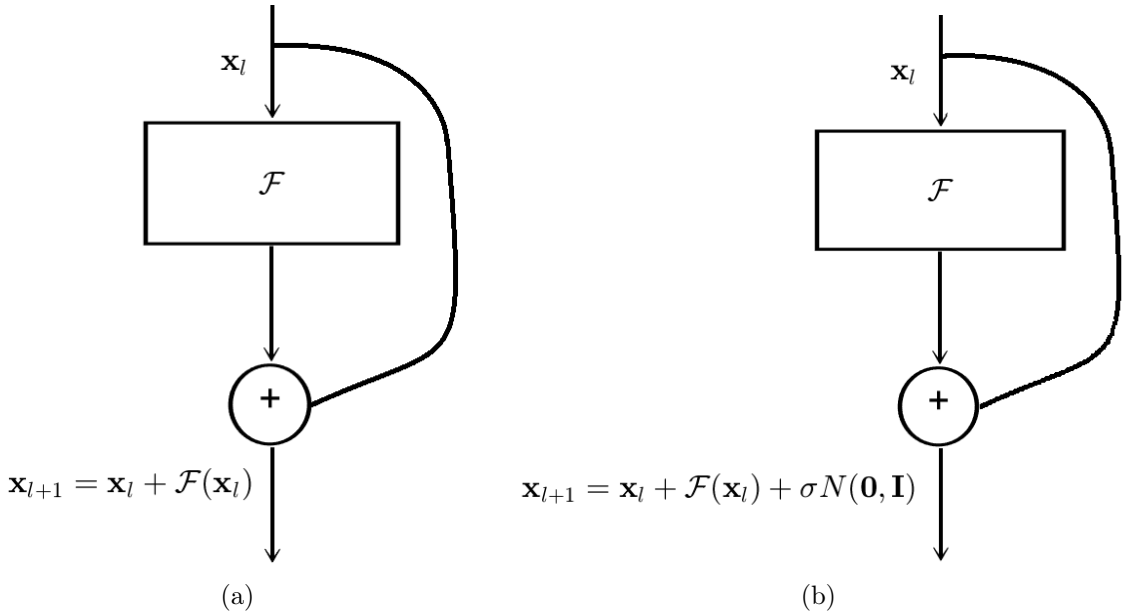


Figure 1: (a) Residual mapping of the ResNet. (b) Gaussian noise injected residual mapping with σ being the variance.

1.2 Related Work

There is a massive volume of research over the last several years on defending against adversarial attacks for DNNs. Randomized smoothing transforms an arbitrary classifier f into a "smoothed" surrogate classifier g and is certifiably robust in ℓ_2 norm based adversarial attacks [34, 33, 12, 54, 8]. Among the randomized smoothing, one of the most popular ideas is to inject Gaussian noise to the input image and the classification result is based on the probability of the noisy image in the decision region. Our adversarial defense algorithm injects noise into each residual mapping instead of the input image, which is different from randomized smoothing.

Robust optimization for solving EARM achieves great success in defending against inference attacks [38, 44, 45, 55, 47]. Regularization in EARM can further boost the robustness of the adversarially trained models [57, 30, 46, 60]. The adversarial defense algorithms should learn a classifier with high test accuracy on both clean and adversarial data. To achieve this goal, Zhang et al. [59] developed a new loss function, TRADES, that explicitly trades off between natural and robust generalization. To the best our of knowledge, TRADES is the current state-of-the-art in defending against inference attacks on the CIFAR10. Throughout this paper, we regard TRADES as the benchmark.

Modeling DNNs as ordinary differential equations (ODEs) has drawn lots of attention recently. Chen et al. proposed neural ODEs for DL [10]. E [14] modeled training ResNets as solving an ODE optimal control problem. Haber and Ruthotto [19] constructed stable DNN architectures based on the properties of numerical ODEs. Lu, Zhu and et al. [37, 61] constructed novel architectures for DNNs, which were motivated from the numerical discretization schemes for ODEs. Sun et al. [50] modeled training of ResNets as solving a stochastic differential equation.

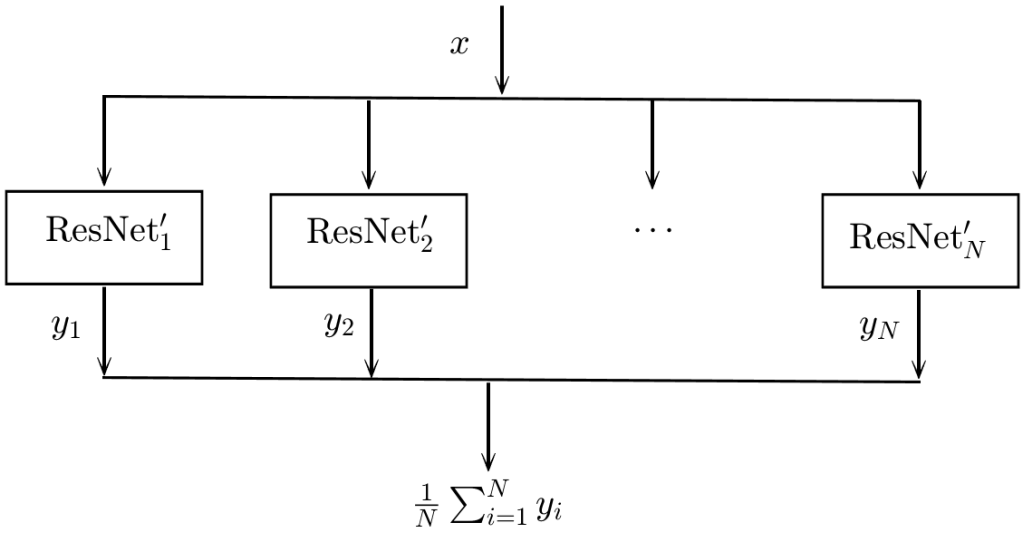


Figure 2: Architecture of the EnResNet.

Model averaging with multiple stochastically trained identical DNNs is the most straightforward ensemble technique to improve the predictive power of base DNNs. This simple averaging method has been a success in image classification for ILSVRC competitions. Different groups of researchers use model averaging for different base DNNs and won different ILSVRC competitions [29, 48, 21]. This widely used unweighted averaging ensemble, however, is not data-adaptive and is sensitive to the presence of excessively biased base learners. Ju et al., recently investigated ensemble of DNNs by many different ensemble methods, including unweighted averaging, majority voting, the Bayes Optimal Classifier, and the (discrete) Super Learner, for image recognition tasks. They concluded that the Super Learner achieves the best performance among all the studied ensemble algorithms [25].

Our work distinguishes from the existing work on DNN ensemble and feature and input smoothing from two major points: First, we inject Gaussian noise to each residual mapping in the ResNet. Second, we jointly train each component of the ensemble instead of using a sequential training.

1.3 Organization

We organize this paper in the following way: In section 2, we model the ResNet as a TE and give an explanation for ResNet's adversarial vulnerability. In section 3, we present a new ResNet ensemble algorithm that motivated from the Feynman-Kac formula for adversarial defense. In section 4, we present the natural accuracy of the EnResNets and their robust accuracy under both white-box and blind PGD and C&W attacks, and compare with the current state-of-the-art. In section 5, we generalize the algorithm to ensemble of different neural nets and numerically verify its efficacy. Our paper ends up with some concluding remarks.

2 Theoretical Motivation and Guarantees

2.1 Transport Equation Modeling of ResNets

The connection between training ResNet and solving optimal control problems of the TE is investigated in [52, 53, 35]. In this section, we derive the TE model for ResNet and explain its adversarial vulnerability from a PDE viewpoint. The TE model enables us to understand the data flow of the entire training and testing data in both forward and backward propagation in training and testing of ResNets; whereas, the ODE models focus on the dynamics of individual data points [10].

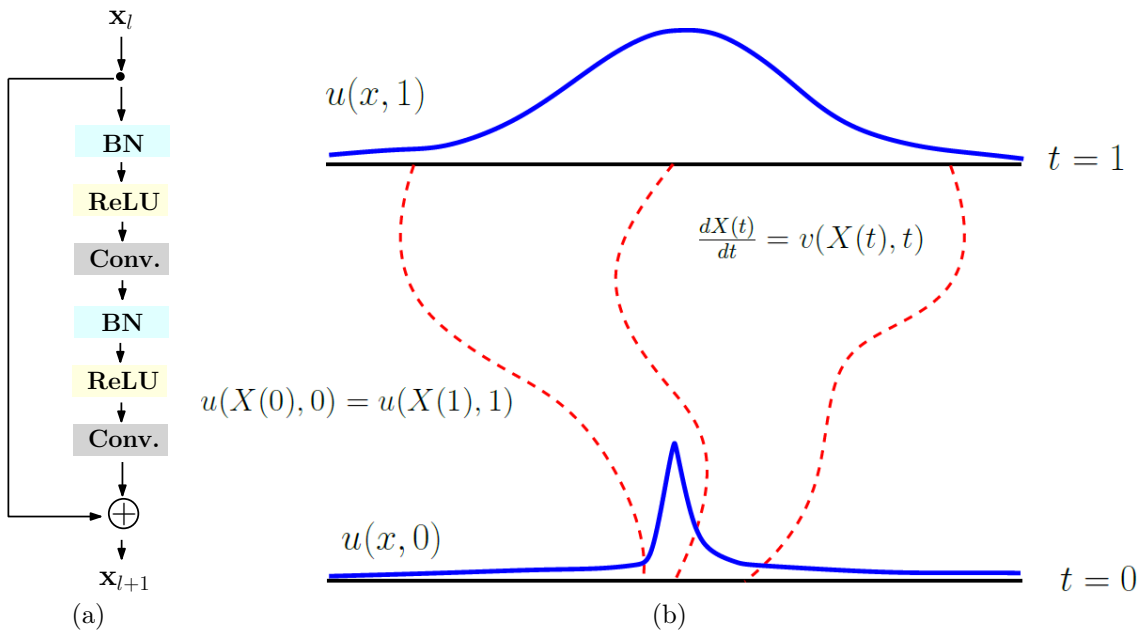


Figure 3: (a) A detailed structure of the residual mapping in the pre-activated ResNet. (b) Demonstration of characteristic curves of the transport equation.

As shown in Fig. 1 (a), residual mapping adds a skip connection to connect the input and output of the original mapping ($\mathcal{F}$), and the l-th residual mapping can be written as

$$\mathbf{x}_{l+1} = \mathcal{F}(\mathbf{x}_l, \mathbf{w}_l) + \mathbf{x}_l,$$

with $\mathbf{x}_0 = \hat{\mathbf{x}} \in T \subset \mathbb{R}^d$ being a data point in the set T, $\mathbf{x}_l$ and $\mathbf{x}_{l+1}$ are the input and output tensors of the residual mapping. The parameters $\mathbf{w}_l$ can be learned by back-propagating the training error. For $\forall\ \hat{\mathbf{x}} \in T$ with label y, the forward propagation of ResNet can be written as

$$\begin{cases} \mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l, \mathbf{w}_l), \quad l = 0, 1, \ldots, L-1, \quad \text{with} \quad \mathbf{x}_0 = \hat{\mathbf{x}}, \\ \hat{y} \doteq f(\mathbf{x}_L), \end{cases} \tag{3}$$

where $\hat{y}$ is the predicted label, L is the number of layers, and $f(\mathbf{x}) = \text{softmax}(\mathbf{w}_0 \cdot \mathbf{x})$ be the output activation with $\mathbf{w}_0$ being the trainable parameters. For the widely used residual mapping in the pre-activated ResNet [22], as shown in Fig. 3 (a), we have

$$\mathcal{F}(\mathbf{x}_l, \mathbf{w}_l) = \mathbf{w}_l^{\text{C2}} \otimes \sigma(\mathbf{w}_l^{\text{B2}} \odot \mathbf{w}_l^{\text{C1}} \otimes \sigma(\mathbf{w}_l^{\text{B1}} \odot x_l)), \tag{4}$$

where $\mathbf{w}_l^{\text{C1}}(\mathbf{w}_l^{\text{B1}})$ and $\mathbf{w}_l^{\text{C2}}(\mathbf{w}_l^{\text{B2}})$ are the first and second convolutional (batch normalization) layers of the l-th residual mapping, respectively, from top to bottom order. $\otimes$ and $\odot$ are the convolutional and batch normalization operators, respectively.

Next, we introduce a temporal partition: let $t_l = l/L$, for $l = 0, 1, \cdots, L$, with the time interval $\Delta t = 1/L$. Without considering dimensional consistency, we regard $\mathbf{x}_l$ in Eq. (3) as the value of $\mathbf{x}(t)$ at the time slot t_l, so Eq. (3) can be rewritten as

$$\begin{cases} \mathbf{x}(t_{l+1}) = \mathbf{x}(t_l) + \Delta t \cdot \overline{F}(\mathbf{x}(t_l), \mathbf{w}(t_l)), \quad l = 0, 1, \ldots, L-1, \quad \text{with} \quad \mathbf{x}(0) = \hat{\mathbf{x}} \\ \hat{y} \doteq f(\mathbf{x}(1)), \end{cases} \tag{5}$$

where $\overline{F} \doteq \frac{1}{\Delta t}\mathcal{F}$. Eq. (5) is the forward Euler discretization of the following ODE

$$\frac{d\mathbf{x}(t)}{dt} = \overline{F}(\mathbf{x}(t), \mathbf{w}(t)), \quad \mathbf{x}(0) = \hat{\mathbf{x}}. \tag{6}$$

Let $u(\mathbf{x}, t)$ be a function that is constant along the trajectory defined by Eq. (6), as demonstrated in Fig. 3 (b), then $u(\mathbf{x}, t)$ satisfies the following TE

$$\frac{d}{dt}\left(u(\mathbf{x}(t), t)\right) = \frac{\partial u}{\partial t}(\mathbf{x}, t) + \overline{F}(\mathbf{x}, \mathbf{w}(t)) \cdot \nabla u(\mathbf{x}, t) = 0, \quad \mathbf{x} \in \mathbb{R}^d, \tag{7}$$

the first equality is because of the chain rule and the second equality dues to the fact that u is constant along the curve defined by Eq. (6).

If we enforce the terminal condition at $t = 1$ for Eq. (7) to be

$$u(\mathbf{x}, 1) = \text{softmax}(\mathbf{w}_0 \cdot \mathbf{x}) := f(\mathbf{x}),$$

then according to the fact that $u(\mathbf{x}, t)$ is constant along the curve defined by Eq. (6) (which is called the characteristic curve for the TE defined in Eq. (7)), we have $u(\hat{\mathbf{x}}, 0) = u(\mathbf{x}(1), 1) = f(\mathbf{x}(1))$; therefore, the forward propagation of ResNet for $\hat{\mathbf{x}}$ can be modeled as computing $u(\hat{\mathbf{x}}, 0)$ along the characteristic curve of the following TE

$$\begin{cases} \frac{\partial u}{\partial t}(\mathbf{x}, t) + \overline{F}(\mathbf{x}, \mathbf{w}(t)) \cdot \nabla u(\mathbf{x}, t) = 0, \quad \mathbf{x} \in \mathbb{R}^d, \\ u(\mathbf{x}, 1) = f(\mathbf{x}). \end{cases} \tag{8}$$

Meanwhile, the backpropagation in training ResNets can be modeled as finding the velocity field, $\overline{F}(\mathbf{x}(t), \mathbf{w}(t))$, for the following control problem

$$\begin{cases} \frac{\partial u}{\partial t}(\mathbf{x}, t) + \overline{F}(\mathbf{x}, \mathbf{w}(t)) \cdot \nabla u(\mathbf{x}, t) = 0, \quad \mathbf{x} \in \mathbb{R}^d, \\ u(\mathbf{x}, 1) = f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^d, \\ u(\mathbf{x}_i, 0) = y_i, \quad \mathbf{x}_i \in T, \quad \text{with } T, \end{cases} \tag{9}$$

where T is the training set that enforces the initial condition on the training data for the TE. Note that in the above TE formulation of ResNet, $u(\mathbf{x}, 0)$ serves as the classifier and the velocity field $\overline{F}(\mathbf{x}, \mathbf{w}(t))$ encodes ResNet's architecture and weights. When $\overline{F}$ is very complex, $u(\mathbf{x}, 0)$ might be highly irregular i.e. a small change in the input $\mathbf{x}$ can lead to a massive change in the value of $u(\mathbf{x}, 0)$. This irregular function may have a good generalizability on clean images, but it is not robust to adversarial attacks. Fig. 4 (a) shows a 2D illustration of $u(\mathbf{x}, 0)$ with the terminal condition $u(\mathbf{x}, 1)$ shown in Fig. 4 (d); we will discuss this in detail later in this section.

2.2 Improving Robustness via Diffusion

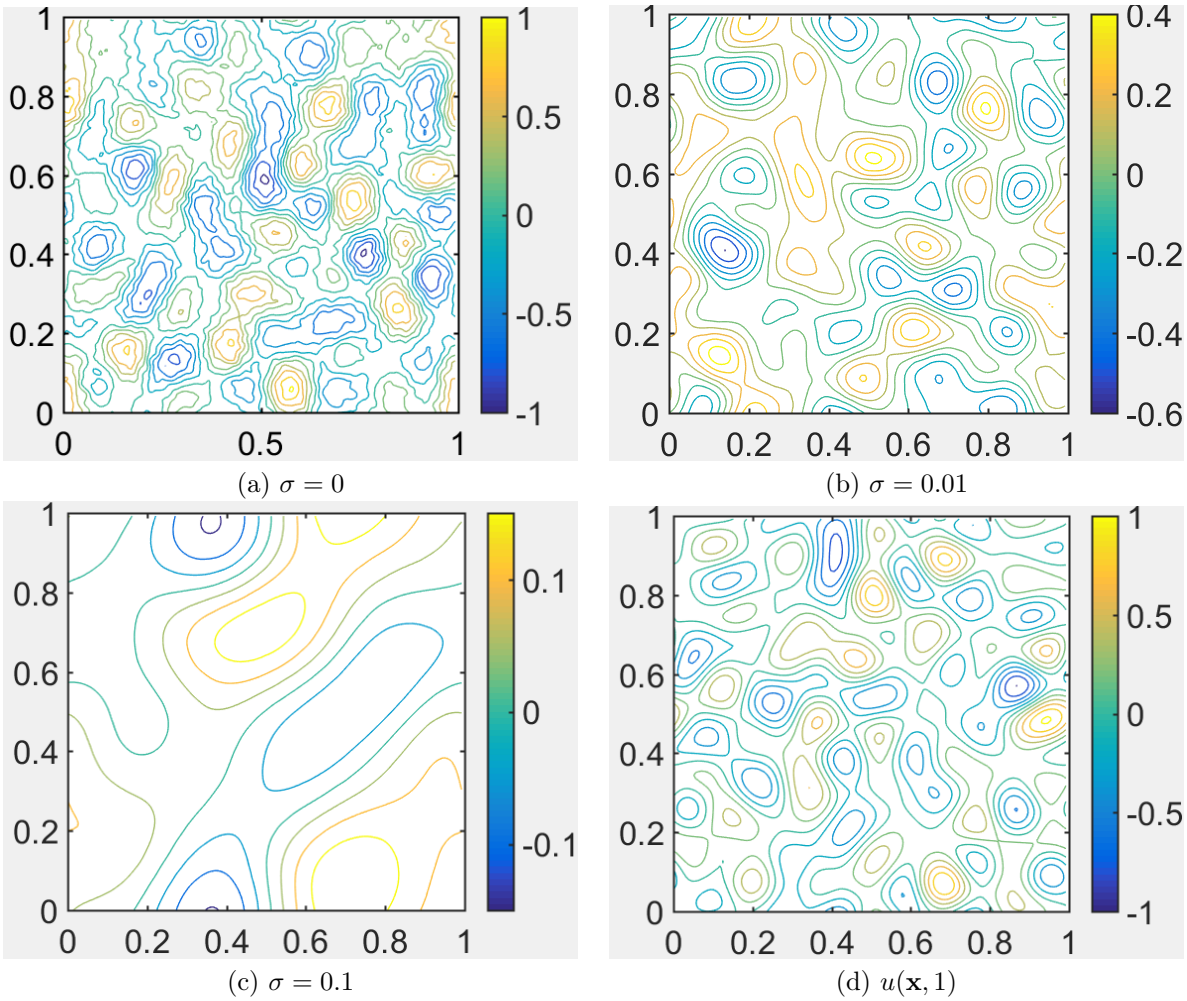
Using a specific level set of $u(\mathbf{x}, 0)$ in Fig. 4 (a) for classification suffers from adversarial vulnerability: A tiny perturbation in $\mathbf{x}$ will lead the output to go across the level set, thus leading to misclassification. To mitigate this issue, we introduce a diffusion term $\frac{1}{2}\sigma^2 \Delta u$ to Eq. (8), with σ being the diffusion coefficient and

$$\Delta = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \cdots + \frac{\partial^2}{\partial x_d^2},$$

is the Laplace operator in $\mathbb{R}^d$. The newly introduced diffusion term makes the level sets of the TE more regular. This improves adversarial robustness of the classifier. Hence, we arrive at the following convection-diffusion equation

$$\begin{cases} \frac{\partial u}{\partial t}(\mathbf{x}, t) + \overline{F}(\mathbf{x}, \mathbf{w}(t)) \cdot \nabla u(\mathbf{x}, t) + \frac{1}{2}\sigma^2 \Delta u(\mathbf{x}, t) = 0, \quad \mathbf{x} \in \mathbb{R}^d, \quad t \in [0, 1), \\ u(\mathbf{x}, 1) = f(\mathbf{x}). \end{cases} \tag{10}$$

The solution of Eq. (10) is much more regular when $\sigma \neq 0$ than when $\sigma = 0$. We consider the solution of Eq. (10) in a 2D unit square with periodic boundary conditions, and on each grid point of the mesh the velocity field $\overline{F}(\mathbf{x}, \mathbf{w}(t))$ is a random number sampled uniformly from -1 to 1. The terminal condition is



(a) $\sigma = 0$ (b) $\sigma = 0.01$

(c) $\sigma = 0.1$ (d) $u(\mathbf{x}, 1)$

Figure 4: (d): terminal condition for Eq. (10); (a), (b), and (c): solutions of the convection-diffusion equation, Eq. (10), at $t = 0$ with different diffusion coefficients σ.

also randomly generated, as shown in Fig. 4 (d). This 2D convection-diffusion equation is solved by the pseudo-spectral method with spatial and temporal step sizes being $1/128$ and 1×10^{-3}, respectively. Figure 4 (a), (b), and (c) illustrate the solutions when $\sigma = 0$, 0.01, and 0.1, respectively. These show that as σ increases, the solution becomes more regular, which makes the classifier more robust, but might be less accurate on clean data. The σ should be selected to have a good trade-off between accuracy and robustness. According to the above observation, instead of using $u(\mathbf{x}, 0)$ of the TE's solution for classification, we use that of the convection-diffusion equation.

2.3 Theoretical Guarantees for the Surrogate Model

We have the following theoretical guarantee for robustness of the solution of the convection-diffusion equation mentioned above.

Theorem 1. *[31] Let $\overline{F}(\mathbf{x}, t)$ be a Lipschitz function in both $\mathbf{x}$ and t, and $f(\mathbf{x})$ be a bounded function. Consider the following initial value problem of the convection-diffusion equation ($\sigma \neq 0$)*

$$\begin{cases} \frac{\partial u}{\partial t}(\mathbf{x}, t) + \overline{F}(\mathbf{x}, \mathbf{w}(t)) \cdot \nabla u(\mathbf{x}, t) + \frac{1}{2}\sigma^2 \Delta u(\mathbf{x}, t) = 0, \quad \mathbf{x} \in \mathbb{R}^d, \quad t \in [0, 1), \\ u(\mathbf{x}, 1) = f(\mathbf{x}). \end{cases} \tag{11}$$

Then, for any small perturbation δ, we have $|u(\mathbf{x}+\delta,0)-u(\mathbf{x},0)| \le C\left(\frac{\|\delta\|_2}{\sigma}\right)^{\alpha}$ for some constant $\alpha>0$ if $\sigma \le 1$. Here, $\|\delta\|_2$ is the ℓ_2 norm of δ, and C is a constant that depends on d, $\|f\|_\infty$, and $\|\overline{F}\|_{L^\infty_{\mathbf{x},t}}$. The meaning of notations $\|f\|_\infty$ and $\|\overline{F}\|_{L^\infty_{\mathbf{x},t}}$ can be found in [31].

Furthermore, we have the following bound for the gradient of the solution of the convection-diffusion equation.

Theorem 2. *Let $u_0(\mathbf{x})$ be a compactly supported function and $\overline{F} \in C^1(\mathbf{R}^d \times [0,1])$. For the following initial value problem of the convection-diffusion equation*

$$\begin{cases} \frac{\partial u}{\partial t}(\mathbf{x},t) + \overline{F}(\mathbf{x},\mathbf{w}(t)) \cdot \nabla u(\mathbf{x},t) = \frac{1}{2}\sigma^2 \Delta u(\mathbf{x},t), \quad \mathbf{x} \in \mathbb{R}^d, \quad t \in [0,1), \\ u(\mathbf{x},0) = u_0(\mathbf{x}), \end{cases} \tag{12}$$

we have

$$\|\nabla u(\mathbf{x},1)\|_\infty \le e^{-\sigma^2} e^{\gamma} \left(\|u_0\|_\infty + \|\nabla u_0\|_\infty\right), \tag{13}$$

where γ is a constant depends on $\nabla \overline{F}$.

Proof. Let $w(\mathbf{x},t) = \left(\mu u^2(\mathbf{x},t) + \|\nabla u(\mathbf{x},t)\|^2\right) e^{-2\lambda t}$, where μ and λ are constants which will be defined later. Note that $u^2(\mathbf{x},t)$ satisfies

$$\frac{\partial (u^2)}{\partial t} + \overline{F}(\mathbf{x},t) \cdot \nabla (u^2) = \sigma^2 \Delta (u^2) - 2\sigma^2 \|\nabla u\|^2,$$

and $\|\nabla u\|^2$ satisfies

$$\frac{\partial \|\nabla u\|^2}{\partial t} + \overline{F}(\mathbf{x},t) \cdot \nabla \|\nabla u\|^2 = -2\nabla u \cdot \nabla \overline{F} \cdot \nabla u + \sigma^2 \Delta \|\nabla u\|^2 - 2\sigma^2 \|\nabla \nabla u\|_F^2,$$

therefore,

$$\frac{\partial w}{\partial t} + \overline{F} \cdot \nabla w - \sigma^2 \nabla w = e^{-2\lambda t}\left[-2\lambda(\mu u^2 + \|\nabla u\|^2) - 2\mu\sigma^2 \|\nabla u\|^2 - 2\nabla u \cdot \nabla \overline{F} \cdot \nabla u - 2\sigma^2 \|\nabla \nabla u\|_F^2\right].$$

Next, let $\gamma(\mathbf{x},t) = \min_{\|\xi\|=1} \xi \cdot \nabla \overline{F} \cdot \xi$ and $\gamma = -\min_{\mathbf{x},t} \gamma(\mathbf{x},t)$, then we have

$$Lw := \frac{\partial w}{\partial t} + \overline{F} \cdot \nabla w - \sigma^2 \nabla w \le -2e^{-2\lambda t}\left[\lambda \mu u^2 + (\lambda + \mu\sigma^2 - \gamma)\|\nabla u\|^2\right].$$

If we choose λ and μ large enough, such that $\lambda + \mu\sigma - \gamma \ge 0$, then

$$Lw \le 0.$$

From the maximum principle, we know $\max_{\mathbf{x}} w(\mathbf{x},1) \le \max_{\mathbf{x}} w(\mathbf{x},0)$, i.e.,

$$\max_{\mathbf{x}} e^{-2\lambda}\left(\mu u^2(\mathbf{x},1) + \|\nabla u(\mathbf{x},1)\|^2\right) \le \max_{\mathbf{x}} \left(\mu u^2(\mathbf{x},0) + \|\nabla u(\mathbf{x},0)\|^2\right).$$

Hence,

$$\|\nabla u(\mathbf{x},1)\|_\infty^2 \le e^{2\lambda}\left(\mu \|u_0\|_\infty^2 + \|\nabla u_0\|_\infty^2\right).$$

Let $\mu = 1$ and $\lambda = \gamma - \sigma^2$, we have

$$\|\nabla u(\mathbf{x},1)\|_\infty \le e^{-\sigma^2}\left(e^{\gamma}\|u_0\|_\infty + e^{\gamma}\|\nabla u_0\|_\infty\right).$$

□

Remark 1. *Similar estimate in Theorem 2 can be established on $u(\mathbf{x},0)$ for the terminal value problem of the convection diffusion equation in Eq. (11) by reverse time.*

3 Algorithms

3.1 ResNets Ensemble via the Feynman-Kac Formula

Based on the above discussion, if we use the solution of the convection-diffusion equation, Eq. (10), for classification. The resulted classifier will be more resistant to adversarial attacks. In this part, we will present an ensemble of ResNets to approximate the solution of Eq. (10). In the Section. 4, we will verify that the robustly trained special ensemble of ResNets is more accurate on both clean and adversarial images than standard ResNets.

The convection-diffusion equation, Eq. (10), can be solved using the Feynman-Kac formula [26] in high dimensional space, which gives $u(\hat{\mathbf{x}}, 0)$ as

$$u(\hat{\mathbf{x}}, 0) = \mathbb{E}\left[f(\mathbf{x}(1))|\mathbf{x}(0) = \hat{\mathbf{x}}\right], \tag{14}$$

where $\mathbf{x}(t)$ is an Itô process,

$$d\mathbf{x}(t) = \overline{F}(\mathbf{x}(t), \mathbf{w}(t))dt + \sigma dB_t,$$

and $u(\hat{\mathbf{x}}, 0)$ is the conditional expectation of $f(\mathbf{x}(1))$.

Next, we approximate the Feynman-Kac formula by an ensemble of modified ResNets in the following way: Accoding to the Euler-Maruyama method [2], the term σdB_t in the Itô process that can be approximated by adding a specially designed Gaussian noise, $\sigma\mathcal{N}(\mathbf{0}, \mathbf{I})$, where $\sigma = a\sqrt{\mathrm{Var}(\mathbf{x}_l + \mathcal{F}(\mathbf{x}_l))}$ with a being a tunable parameter, to each original residual mapping $\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l)$ in the ResNet. This gives the modified residual mapping $\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l) + \sigma\mathcal{N}(\mathbf{0}, \mathbf{I})$, as illustrated in Fig. 1 (b). Let ResNet' denote the modified ResNet where we inject noise to each residual mapping of the original ResNet. In a nutshell, ResNet's approximation to the Feynman-Kac formula is an ensemble of jointly trained ResNet' as illustrated in Fig. 1 (c). [1] We call this ensemble of ResNets as EnResNet. For instance, if the base ResNet is ResNet20, an ensemble of n ResNet20 is denoted as En$_n$ResNet20.

3.2 Adversarial Attacks

In this subsection, we review a few widely used adversarial attacks. These attacks will be used to train robust EnResNets and attack the trained models. We attack the trained model, $f(\mathbf{x}, \mathbf{w})$, by ℓ_∞ norm based (the other norm based attacks can be formulated similarly) untargeted fast gradient sign method (FGSM), iterative FGSM (IFGSM) [16], and Carlini-Wagner (C&W) [9] attacks in both white-box and blind fashions. In blind attacks, we use the target model to classify the adversarial images crafted by attacking the oracle model in a white-box approach. For a given instance $(\mathbf{x}, y)$:

- FGSM searches the adversarial image $\mathbf{x}'$ by maximizing the loss function $\mathcal{L}(\mathbf{x}', y) \doteq \mathcal{L}(f(\mathbf{x}', \mathbf{w}), y)$, subject to the constraint $||\mathbf{x}' - \mathbf{x}||_\infty \leq \epsilon$ with ϵ being the maximum perturbation. For the linearized loss function, $\mathcal{L}(\mathbf{x}', y) \approx \mathcal{L}(\mathbf{x}, y) + \nabla_x\mathcal{L}(\mathbf{x}, y)^T \cdot (\mathbf{x}' - \mathbf{x})$, the optimal adversarial is

$$\mathbf{x}' = \mathbf{x} + \epsilon \cdot \mathrm{sign}\left(\nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x}, y)\right). \tag{15}$$

- IFGSM, Eq. (16), iterates FGSM with step size α and clips the perturbed image to generate the enhanced adversarial attack,

$$\mathbf{x}^{(m)} = \mathrm{Clip}_{\mathbf{x},\epsilon}\{\mathbf{x}^{(m-1)} + \alpha \cdot \mathrm{sign}(\nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x}^{(m-1)}, y))\}, \tag{16}$$

  where $m = 1, \cdots, M$, $\mathbf{x}^{(0)} = \mathbf{x}$, and let the adversarial image be $\mathbf{x}' = \mathbf{x}^{(M)}$ with M being the total number of iterations.

- C&W attack searches the targeted adversarial image by solving

$$\min_\delta ||\delta||_\infty, \quad \text{subject to} \quad f(\mathbf{w}, \mathbf{x} + \delta) = t, \ \mathbf{x} + \delta \in [0, 1]^d, \tag{17}$$

[1] To ease the notation, in what follows, we use ResNet in place of ResNet' when there is no ambiguity.

where δ is the adversarial perturbation and t is the target label. Carlini et al. [9] proposed the following approximation to Eq. (17),

$$\min_{\mathbf{u}} ||\frac{1}{2}(\tanh(\mathbf{u})+1)-\mathbf{x}||_\infty + \tag{18}$$
$$c\cdot\max\left\{-\kappa, \max_{i\neq t}(Z(\frac{1}{2}(\tanh(\mathbf{u}))+1)_i) - Z(\frac{1}{2}(\tanh(\mathbf{u}))+1)_t\right\},$$

where $Z(\cdot)$ is the logit vector for the input, i.e., the output of the DNN before the softmax layer. This unconstrained optimization problem can be solved efficiently by using the Adam optimizer [27]. Dou et al. [13], prove that, under a certain regime, C&W can shift the DNNs' predicted probability distribution to the desired one.

All three attacks clip the pixel values of the adversarial image to between 0 and 1. In the following experiments, we set $\epsilon = 8/255$ in both FGSM and IFGSM attacks. Additionally, in IFGSM we set $m = 20$ and $\alpha = 2/255$, and denote it as IFGSM^{20}. For C&W attack, we run 50 iterations of Adam with learning rate 6×10^{-4} and set $c = 10$ and $\kappa = 0$.

3.3 Robust Training of EnResNets

We use the PGD adversarial training [38], i.e., solving EARM Eq. (1) by replacing $\mathbf{x}'$ with the PGD adversarial one, to robustly train EnResNets with $\sigma = 0.1$ on both CIFAR10 and CIFAR100 [28] benchmarks with standard data augmentation [21]. The attack in the PGD adversarial training is merely IFGSM with an initial random perturbation on the clean data. We summarize the PGD based robust training for EnResNets in Algorithm 1. Other methods to solve EARM can also be used to train EnResNets, e.g., approximation to the adversarial risk function and regularization. EnResNet enriches the hypothesis class $\mathcal{H}$, to make the classifiers from $\mathcal{H}$ more adversarially robust. All computations are carried out on a machine with a single Nvidia Titan Xp graphics card.

Algorithm 1 Training of the EnResNet by PGD Adversarial Training

Input: Training set: $(\mathbf{X}_i, \mathbf{Y}_i)_{i=1}^{N_B}$, N_B = #minibatches, perturbation ϵ, and step size α.
Output: A robustly trained En$_N$ResNet, i.e., an ensemble of N modified ResNets.
for $i = 1, \ldots, N_E$ (where N_E is the number of epochs.) **do**
  for $j = 1, \ldots, N_B$ **do**
    //PGD attack
    Add uniform noise in the range $[-\epsilon, \epsilon]$ to $\mathbf{X}_i$, denote the resulted images as $\tilde{\mathbf{X}}_i$.
    Attack $\tilde{\mathbf{X}}_i$ by 10 iterations IFGSM attacks with maximum perturbation ϵ and step size α. And denote the adversarial images as $\mathbf{X}'_i$.
    //Forward-propagation
    Generate prediction $\tilde{\mathbf{Y}}_i = \text{En}_N\text{ResNet}(\mathbf{X}'_i)$ for $\mathbf{X}'_i$ by the current model En$_N$ResNet.
    //Back-propagation
    Back-Propagate the cross-entropy loss between $\mathbf{Y}_i$ and $\tilde{\mathbf{Y}}_i$ to update the model EnResNet$_N$.

4 Numerical Results

In this section, we numerically verify that the robustly trained EnResNets are more accurate, on both clean and adversarial data of the CIFAR10 and CIFAR100, than robustly trained ResNets and ensemble of ResNets without noise injection. To avoid the gradient mask issue of EnResNets due to the noise injection in each residual mapping, we use the Expectation over Transformation (EOT) strategy [6] to compute the gradient which is averaged over five independent runs.

Table 1: Natural accuracies of naturally trained ResNet20 and different ensemble of noise injected ResNet20 on the CIFAR10 dataset. Unit: %.

Model	dataset	$\mathcal{A}_{\text{nat}}$
ResNet20	CIFAR10	92.10
En_1ResNet20	CIFAR10	92.59
En_2ResNet20	CIFAR10	92.60
En_5ResNet20	CIFAR10	92.74
ResNet44	CIFAR10	93.22
En_1ResNet44	CIFAR10	93.37
En_2ResNet44	CIFAR10	93.54
ResNet110	CIFAR10	94.30
En_2ResNet110	CIFAR10	93.49

4.1 Natural and Robust Accuracies of Robustly Trained EnResNets

In robust training, we run 200 epochs of the PGD adversarial training (10 iterations of IFGSM with $\alpha = 2/255$ and $\epsilon = 8/255$, and an initial random perturbation of magnitude ϵ) with initial learning rate 0.1, which decays by a factor of 10 at the 80th, 120th, and 160th epochs. The training data is split into 45K/5K for training and validation, the model with the best validation accuracy is used for testing. Similar settings are used for natural training, i.e., solving the ERM problem Eq. (2). En_1ResNet20 denotes the ensemble of only one ResNet20 which is merely adding noise to each residual mapping, and similar notations apply to other DNNs.

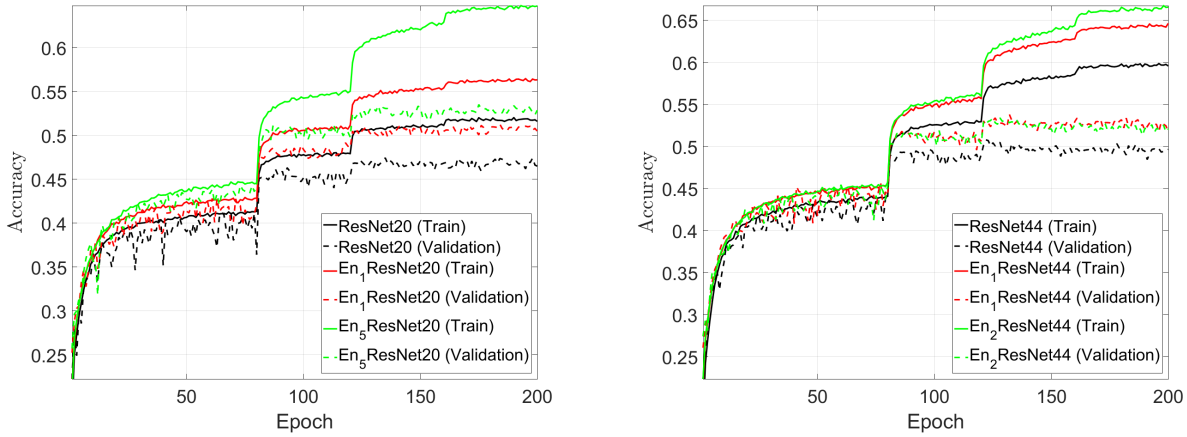
First, we show that the ensemble of noise injected ResNets can improve the natural generalization of the naturally trained models. As shown in Table 1, the naturally trained ensemble of multiple ResNets are always generalize better on the clean images than the base ResNets. This conclusion is verified by ResNet20, ResNet44, and ResNet110. However, the natural accuracy of the robustly trained models are much less than that of the naturally trained models. For instance, the natural accuracies of the robustly trained and naturally trained ResNet20 are, respectively, 75.11% and 92.10%. The degradation of natural accuracies in robust training are also confirmed by experiments on ResNet44 (78.89% v.s. 93.22%) and ResNet110 (82.19% v.s. 94.30%). Improving natural accuracy of the robustly trained models is another important issue during adversarial defense.

Second, consider natural ($\mathcal{A}_{\text{nat}}$) and robust ($\mathcal{A}_{\text{rob}}$) accuracies of the PGD adversarially trained models on the CIFAR10, where $\mathcal{A}_{\text{nat}}$ and $\mathcal{A}_{\text{rob}}$ are measured on clean and adversarial images, respectively. All results are listed in Table 2. The robustly trained ResNet20 has accuracies 50.89%, 46.03% (close to that reported in [38]), and 58.73%, respectively, under the FGSM, IFGSM^{20}, and C&W attacks. Moreover, it has a natural accuracy of 75.11%. En_5ResNet20 boosts natural accuracy to 82.52%, and improves the corresponding robust accuracies to 58.92%, 51.48%, and 67.73%, respectively. Simply injecting noise to each residual mapping of ResNet20 can increase $\mathcal{A}_{\text{nat}}$ by $\sim 2\%$ and $\mathcal{A}_{\text{rob}}$ by $\sim 3\%$ under the IFGSM^{20} attack. The advantages of EnResNets are also verified by experiments on ResNet44, ResNet110, and their ensembles. Note that ensemble of high capacity ResNet is more robust than low capacity model: as shown in Table 2, En_2ResNet110 is more accurate than En_2ResNet44 which in turn is more accurate than En_2ResNet20 in classifying both clean and adversarial images. The robustly trained En_1WideResNet34-10 has 86.19% and 56.60%, respectively, natural and robust accuracies under the IFGSM^{20} attack. Compared with the current state-of-the-art [59], En_1WideResNet34-10 has almost the same robust accuracy (56.60% v.s. 56.61%) under the IFGSM^{20} attack but better natural accuracy (86.19% v.s. 84.92%). Figure 5 plots the evolution of training and validation accuracies of ResNet20 and ResNet44 and their different ensembles.

Third, consider accuracy of the robustly trained models under blind attacks. In this scenario, we use the target model to classify the adversarial images crafted by applying FGSM, IFGSM^{20}, and C&W attacks to the oracle model. As listed in Table 3, EnResNets are always more robust than the base ResNets under different blind attacks. For instance, when En_5ResNet20 is used to classify adversarial images crafted by attacking ResNet20 with FGSM, IFGSM^{20}, and C&W attacks, the accuracies are 64.07%, 62.99%, and 76.57%, respectively. Conversely, the accuracies of ResNet20 are only 61.69%, 58.74%, and 73.77%, respectively, in classifying adversarial images obtained by using the above three attacks to attack En_5ResNet20.

Table 2: Natural and robust accuracies of different base and noise injected ensembles of robustly trained ResNets on the CIFAR10. Unit: %.

Model	dataset	$\mathcal{A}_{\text{nat}}$	$\mathcal{A}_{\text{rob}}$ (FGSM)	$\mathcal{A}_{\text{rob}}$ (IFGSM^{20})	$\mathcal{A}_{\text{rob}}$ (C&W)
ResNet20	CIFAR10	75.11	50.89	46.03	58.73
En$_1$ResNet20	CIFAR10	77.21	55.35	49.06	65.69
En$_2$ResNet20	CIFAR10	80.34	57.23	50.06	66.47
En$_5$ResNet20	CIFAR10	82.52	58.92	51.48	67.73
ResNet44	CIFAR10	78.89	54.54	48.85	61.33
En$_1$ResNet44	CIFAR10	82.03	57.80	51.83	66.00
En$_2$ResNet44	CIFAR10	82.91	58.29	51.86	66.89
ResNet110	CIFAR10	82.19	57.61	52.02	62.92
En$_2$ResNet110	CIFAR10	82.43	59.24	53.03	68.67
En$_1$WideResNet34-10	CIFAR10	**86.19**	**61.82**	**56.60**	**69.32**



(a) (b)

Figure 5: Evolution of training and validation accuracy. (a): ResNet20 and different ensembles of noise injected ResNet20. (b): ResNet44 and different ensembles of noise injected ResNet44.

Table 3: Accuracies of robustly trained models on adversarial images of CIFAR10 crafted by attacking the oracle model with different attacks. Unit: %.

Model	dataset	Oracle	$\mathcal{A}_{\text{rob}}$ (FGSM)	$\mathcal{A}_{\text{rob}}$ (IFGSM^{20})	$\mathcal{A}_{\text{rob}}$ (C&W)
ResNet20	CIFAR10	En$_5$ResNet20	61.69	58.74	73.77
En$_5$ResNet20	CIFAR10	ResNet20	64.07	62.99	76.57
ResNet44	CIFAR10	En$_2$ResNet44	63.87	60.66	75.83
En$_2$ResNet44	CIFAR10	ResNet44	64.52	61.23	76.99
ResNet110	CIFAR10	En$_2$ResNet110	64.19	61.80	75.19
En$_2$ResNet110	CIFAR10	ResNet110	66.26	62.89	77.71

Fourth, we perform experiments on the CIFAR100 to further verify the efficiency of EnResNets in defending against adversarial attacks. Table 4 lists the naturally accuracies of the naturally trained ResNets and their ensembles, again, the ensemble can improve natural accuracies. Table 5 lists natural and robust accuracies of robustly trained ResNet20, ResNet44, and their ensembles under white-box attacks. The robust accuracy under the blind attacks is listed in Table 6. The natural accuracy of the PGD adversarially trained baseline ResNet20 is 46.02%, and it has robust accuracies 24.77%, 23.23%, and 32.42% under FGSM, IFGSM^{20}, and C&W attacks, respectively. En$_5$ResNet20 increases them to 51.72%, 31.64%, 27.80%, and 40.44%, respectively. The ensemble of ResNets is more effective in defending against adversarial attacks than making the ResNets deeper. For instance, En$_2$ResNet20 that has $\sim 0.27M \times 2$ parameters is much more robust to adversarial attacks, FGSM (30.20% v.s. 28.40%), IFGSM^{20} (26.25% v.s. 25.81%), and C&W (40.06% v.s. 36.06%), than

Table 4: Natural accuracies of naturally trained ResNet20 and different ensemble of noise injected ResNet20 on the CIFAR100 dataset. Unit: %.

Model	dataset	$\mathcal{A}_{\text{nat}}$
ResNet20	CIFAR100	68.53
ResNet44	CIFAR100	71.48
En_2ResNet20	CIFAR100	69.57
En_5ResNet20	CIFAR100	70.22

ResNet44 with $\sim 0.66M$ parameters. Under blind attacks, En_2ResNet20 is also significantly more robust to different attacks where the opponent model is used to generate adversarial images. Under the same model and computation complexity, EnResNets is more robust to adversarial images and more accurate on clean images than deeper nets.

Table 5: Natural and robust accuracies of robustly trained ResNet20 and different ensemble of noise injected ResNet20 on the CIFAR100. Unit: %.

Model	dataset	$\mathcal{A}_{\text{nat}}$	$\mathcal{A}_{\text{rob}}$ (FGSM)	$\mathcal{A}_{\text{rob}}$ (IFGSM^{20})	$\mathcal{A}_{\text{rob}}$ (C&W)
ResNet20	CIFAR100	46.02	24.77	23.23	32.42
En_2ResNet20	CIFAR100	50.68	30.20	26.25	40.06
En_5ResNet20	CIFAR100	**51.72**	**31.64**	**27.80**	**40.44**
ResNet44	CIFAR100	50.38	28.40	25.81	36.06

Table 6: Accuracies of robustly trained models on the adversarial images of CIFAR100 crafted by attacking the oracle model with different attacks. Unit: %.

Model	dataset	Oracle	$\mathcal{A}_{\text{rob}}$ (FGSM)	$\mathcal{A}_{\text{rob}}$ (IFGSM^{20})	$\mathcal{A}_{\text{rob}}$ (C&W)
ResNet20	CIFAR100	En_2ResNet20	33.08	30.79	41.52
En_2ResNet20	CIFAR100	ResNet20	34.15	33.34	48.21

Figure 6 depicts a few selected images from the CIFAR10 and their adversarial ones crafted by applying either IFGSM^{20} or C&W attack to attack both ResNet20 and En_5ResNet20. Both adversarially trained ResNet20 and En_5ResNet20 fail to correctly classify any of the adversarial versions of these four images. For the deer image, it might also be difficult for human to distinguish it from a horse.

4.2 Integration of Separately Trained EnResNets

In the previous subsection, we verified the adversarial defense capability of EnResNet, which is an approximation to the Feynman-Kac formula to solve the convection-diffusion equation. As we showed, when more ResNets and larger models are involved in the ensemble, both natural and robust accuracies are improved. However, EnResNet proposed above requires to train the ensemble jointly, which poses memory challenges for training ultra-large ensembles. To overcome this issue, we consider training each component of the ensemble individually and integrating them together for prediction. The major benefit of this strategy is that with the same amount of GPU memory, we can train a much larger model for inference since the batch size used in inference can be one.

Table 7 lists natural and robust accuracies of the integration of separately trained EnResNets on the CIFAR10. The integration of separately trained EnResNets have better robust accuracy than each component. For instance, the integration of En_2ResNet110 and En_1WideResNet34-10 gives a robust accuracy $\mathbf{57.94}\%$ under the IFGSM^{20} attack, which is remarkably better than both En_2ResNet110 (53.05%) and En_1WideResNet34-10 (56.60%). To the best of our knowledge, 57.94% outperforms the current state-of-the-art [59] by 1.33%. The effectiveness of the integration of separately trained EnResNets sheds light on the development of ultra-large models to improve efficiency for adversarial defense.

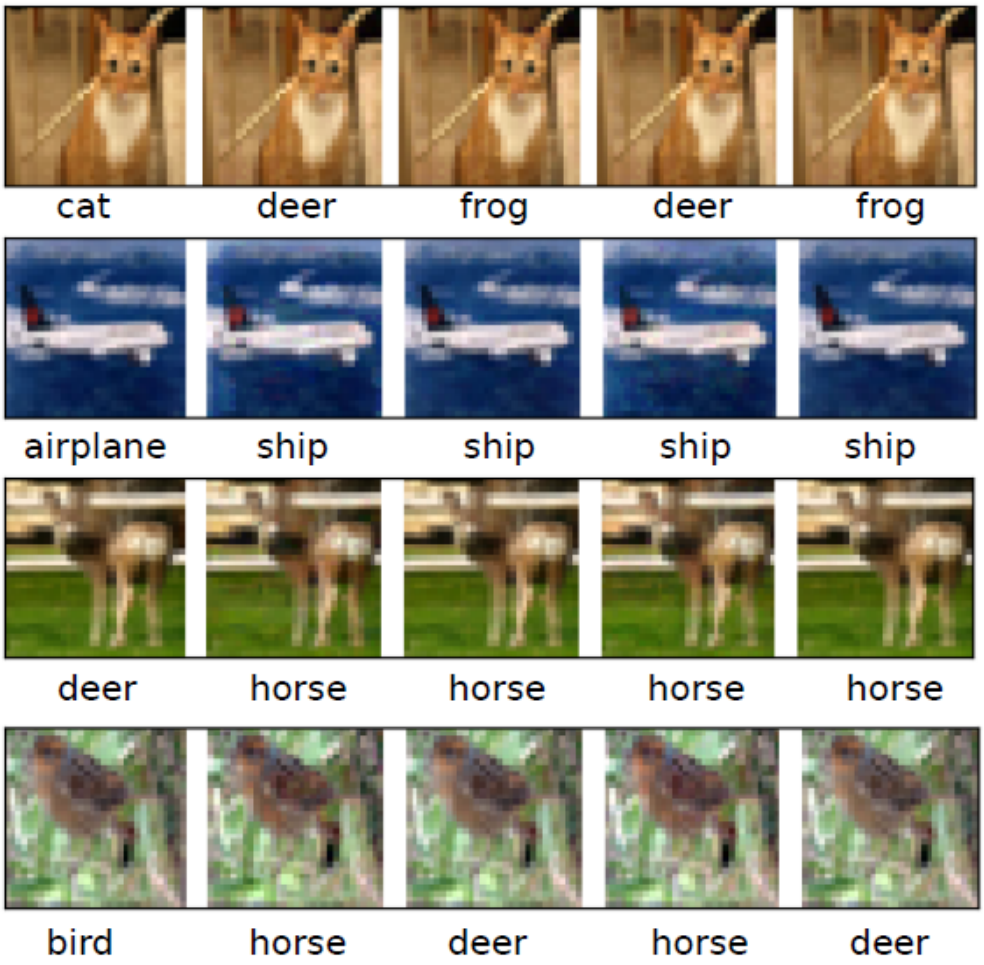


Figure 6: Column 1: original images and labels; column 2-3 (4-5): adversarial images crafted by using IFGSM^{20} and C&W to attack ResNet20 (En$_5$ResNet20) and corresponding predicted labels.

Table 7: Natural and robust accuracies of different integration of different robustly trained EnResNets on the CIFAR10. Unit: %.

Model	dataset	$\mathcal{A}_{\text{nat}}$	$\mathcal{A}_{\text{rob}}$ (FGSM)	$\mathcal{A}_{\text{rob}}$ (IFGSM^{20})	$\mathcal{A}_{\text{rob}}$ (C&W)
En$_2$ResNet20&En$_5$ResNet20	CIFAR10	82.82	59.14	53.15	68.00
En$_2$ResNet44&En$_5$ResNet20	CIFAR10	82.99	59.64	53.86	69.36
En$_2$ResNet110&En$_5$ResNet20	CIFAR10	83.57	60.63	54.87	70.02
En$_2$ResNet110&En$_1$WideResNet34-10	CIFAR10	**85.62**	**62.48**	**57.94**	**70.20**

4.3 Comparison with the Wide ResNet

In this subsection, we show that with the same number of parameters, EnResNets is more adversarially robust that the Wide ResNets. We compare EnResNet$_2$20 with the wide-ResNet: WRN-14-2 [58]. WRN-14-2 has ~ 0.69M parameters which is more than that of EnResNet$_2$20. We list natural and robust accuracies of the robustly trained models on the CIFAR10 benchmark in Table. 8. En$_2$ResNet20 has higher natural accuracy than WRN-14-2 (80.34% v.s. 78.37%). Moreover, En$_2$ResNet20 is more robust to both IFGSM^{20} and C&W attacks.

Table 8: Natural and robust accuracies of robustly trained En2esNet20 and WRN-14-2 on the CIFAR10 dataset. Unit: %.

Model	dataset	$\mathcal{A}_{\text{nat}}$	$\mathcal{A}_{\text{rob}}$ (FGSM)	$\mathcal{A}_{\text{rob}}$ (IFGSM^{20})	$\mathcal{A}_{\text{rob}}$ (C&W)
En_2ResNet20	CIFAR10	80.34	57.23	50.06	66.47
WRN-14-2	CIFAR10	78.37	52.93	48.85	60.30

4.4 Gradient Mask and Comparison with Simple Ensembles

Besides applying EOT gradient, we further verify that our defense is not due to obfuscated gradient. We use IFGSM^{20} to attack naturally trained (using the same approach as that used in [21]) En_1ResNet20, En_2ResNet20, and En_5ResNet20, and the corresponding accuracies are: 0%, 0.02%, and 0.03%, respectively. All naturally trained EnResNets are easily fooled by IFGSM^{20}, thus gradient mask does not play an important role in EnResNets for adversarial defense [5].

Ensemble of models for adversarial defense has been studied in [49]. Here, we show that ensembles of robustly trained ResNets without noise injection cannot boost natural and robust accuracy much. The natural accuracy of jointly (separately) adversarially trained ensemble of two ResNet20 without noise injection is 75.75% (74.96%), which does not substantially outperform ResNet20 with a natural accuracy 75.11%. The corresponding robust accuracies are 51.11% (51.68%), 47.28% (47.86%), and 59.73% (59.80%), respectively, under the FGSM, IFGSM^{20}, and C&W attacks. These robust accuracies are much inferior to that of En_2ResNet20. Furthermore, the ensemble of separately trained robust ResNet20 and robust ResNet44 gives a natural accuracy of 77.92%, and robust accuracies are 54.73%, 51.47%, 61.77% under the above three attacks. These results reveal that ensemble adversarially trained ResNets via the Feynman-Kac formalism is much more accurate than standard ensemble in both natural and robust generalizations.

5 Ensemble of Different ResNets

In previous sections, we proposed and numerically verifies the efficiency of the EnResNet, which can be regarded as an Monte Carlo (MC) approximation to the Feynman-Kac formula that used to solve the convection-diffusion equation. A straightforward extension is to solve the convection-diffusion equation by the multi-level MC [15], which in turn can be simulated by an ensemble of ResNets with different depths. In previous ensembles, we used the same weight for each individual ResNet. However, in the ensemble of different ResNets, we learn the optimal weight for each component. Here, we derive the formula to learn the optimal weights in the cross-entropy loss setting.

Suppose we have an ensemble of two ResNets for n-class classification with training data $\{\mathbf{x}_i, y_i\}_{i=1}^N$ where y_i is the label of $\mathbf{x}_i$ and N is the number of training data. Let the tensors before the softmax output activation of two ResNet, respectively, be

$$\tilde{\mathbf{y}}_i = \left(\tilde{y}_i^1, \tilde{y}_i^2 \ldots, \tilde{y}_i^n\right),$$

and

$$\hat{\mathbf{y}}_i = \left(\hat{y}_i^1, \hat{y}_i^2, \ldots, \hat{y}_i^n\right),$$

where $i = 1, 2, \ldots, N$.

The ensemble of these two ResNets gives the following output before the softmax output activation for the i-th instance

$$\mathbf{y}_i = w_1\tilde{\mathbf{y}}_i + w_2\hat{\mathbf{y}}_i = \begin{pmatrix} w_1\tilde{y}_i^1 + w_2\hat{y}_i^1 \\ w_1\tilde{y}_i^2 + w_2\hat{y}_i^2 \\ \vdots \\ w_1\tilde{y}_i^n + w_2\hat{y}_i^n \end{pmatrix}.$$

Table 9: Natural and robust accuracies of the robustly trained En_2ResNet32 and En_1ResNet20&En_1ResNet32 on the CIFAR10 dataset. Unit: %.

Model	dataset	$\mathcal{A}_{\text{nat}}$	$\mathcal{A}_{\text{rob}}$ (IFGSM^{20})	$\mathcal{A}_{\text{rob}}$ (C&W)
En_2ResNet32	CIFAR10	81.46	52.06	68.41
En_1ResNet20&En_1ResNet32	CIFAR10	81.56	51.99	68.62

Table 10: Natural and robust accuracies of the robustly trained En_2ResNet32 and En_1ResNet20&En_1ResNet32 on the CIFAR100 dataset. Unit: %.

Model	dataset	$\mathcal{A}_{\text{nat}}$	$\mathcal{A}_{\text{rob}}$ (IFGSM^{20})	$\mathcal{A}_{\text{rob}}$ (C&W)
En_2ResNet32	CIFAR100	53.14	27.27	41.50
En_1ResNet20&En_1ResNet32	CIFAR100	53.07	27.01	42.23

where w_1 and w_2 are the weights of the two ResNets, where we enforce $w_1 + w_2 = 1$. Hence, the corresponding log-softmax for the i-th instance is

$$\begin{pmatrix} \log\left(\frac{\exp\left(w_1\tilde{y}_i^1+w_2\hat{y}_i^1\right)}{\sum_{j=1}^n \exp\left(w_1\tilde{y}_i^j+w_2\hat{y}_i^j\right)}\right) \\ \log\left(\frac{\exp\left(w_1\tilde{y}_i^2+w_2\hat{y}_i^2\right)}{\sum_{j=1}^n \exp\left(w_1\tilde{y}_i^j+w_2\hat{y}_i^j\right)}\right) \\ \vdots \\ \log\left(\frac{\exp\left(w_1\tilde{y}_i^n+w_2\hat{y}_i^n\right)}{\sum_{j=1}^n \exp\left(w_1\tilde{y}_i^j+w_2\hat{y}_i^j\right)}\right) \end{pmatrix},$$

Let L be the total cross-entropy loss on these N training data, then we have

$$\frac{\partial L}{\partial w_1} = -\sum_{i=1}^N \left(\tilde{y}_i^{t_i} - \frac{\sum_{j=1}^n \tilde{y}_i^j \exp\left(w_1\tilde{y}_i^j + w_2\hat{y}_i^j\right)}{\sum_{j=1}^n \exp\left(w_1\tilde{y}_i^j + w_2\hat{y}_i^j\right)}\right), \tag{19}$$

and

$$\frac{\partial L}{\partial w_2} = -\sum_{i=1}^N \left(\hat{y}_i^{t_i} - \frac{\sum_{j=1}^n \hat{y}_i^j \exp\left(w_1\tilde{y}_i^j + w_2\hat{y}_i^j\right)}{\sum_{j=1}^n \exp\left(w_1\tilde{y}_i^j + w_2\hat{y}_i^j\right)}\right). \tag{20}$$

In implementation, we update these weights once per epoch during the training and normalize the updated weights.

To show performance of ensembles of jointly trained different ResNets, we robustly train an ensemble of noise injected ResNet20 and ResNet32 on both CIFAR10 and CIFAR100 benchmarks. As shown in Tables 9 and 10, on CIFAR10 the ensemble of jointly trained noise injected ResNet20 and ResNet32 outperforms En_2ResNet32 in classifying both clean (81.46% v.s. 81.56%) and adversarial images of C&W attack (68.41% v.s. 68.62%). On CIFAR100, performances of the ensemble of jointly trained noise injected ResNet20 and ResNet32 and En_2ResNet32 are comparable.

6 Concluding Remarks

Motivated by the transport equation modeling of the ResNet and the Feynman-Kac formula, we proposed a novel ensemble algorithm for ResNets. The proposed ensemble algorithm consists of two components: injecting Gaussian noise to each residual mapping of ResNet, and averaging over multiple jointly and robustly trained baseline ResNets. Numerical results on the CIFAR10 and CIFAR100 show that our ensemble algorithm improves both natural and robust generalization of the robustly trained models. Our approach is a complement to many existing adversarial defenses, e.g., regularization based approaches for adversarial training [59]. It is of interesting to explore the regularization effects in EnResNet.

The memory consumption is one of the major bottlenecks in training ultra-large DNNs. Another advantage of our framework is that we can train small models and integrate them during testing.

Acknowledgments

This material is based on research sponsored by the Air Force Research Laboratory under grant numbers FA9550-18-0167, DARPA FA8750-18-2-0066, and MURI FA9550-18-1-0502, the Office of Naval Research under grant number N00014-18-1-2527, the U.S. Department of Energy under grant number DOE SC0013838, the National Science Foundation under grant number DMS-1554564, (STROBE), and by the Simons foundation. Zuoqiang Shi is supported by NSFC 11671005. Bao Wang thanks Farzin Barekat, Hangjie Ji, Jiajun Tong, and Yuming Zhang for stimulating discussions.

References

[1] Adversarial machine learning against Tesla's autopilot. `https://www.schneier.com/blog/archives/2019/04/adversarial_mac.html`.

[2] P. Kloeden abd E. Platen. Numerical Solution of Stochastic Differential Equations. Springer, 1992.

[3] N. Akhtar and A. Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. arXiv preprint arXiv:1801.00553, 2018.

[4] Naveed Akhtar, Jian Liu, and Ajmal Mian. Defense against universal adversarial perturbations. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[5] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. International Conference on Machine Learning, 2018.

[6] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok. Synthesizing robust adversarial examples. International Conference on Machine Learning, 2018.

[7] W. Brendel, J. Rauber, and M. Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. arXiv preprint arXiv:1712.04248, 2017.

[8] X. Cao and N. Gong. Mitigating evasion attacks to deep neural networks via region-based classification. In 33rd Annual Computer Security Applications Conference, 2017.

[9] N. Carlini and D.A. Wagner. Towards evaluating the robustness of neural networks. IEEE European Symposium on Security and Privacy, pages 39–57, 2016.

[10] R. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations. In Advances in Neural Information Processing Systems, 2018.

[11] X. Chen, C. Liu, B. Li, K. Liu, and D. Song. Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526, 2017.

[12] J. Cohen, E. Rosenfeld, and J.Z. Kolter. Certified adversarial robustness via randomized smoothing. arXiv preprint arXiv:1902.02918v1, 2019.

[13] Z. Dou, S. J. Osher, and B. Wang. Mathematical analysis of adversarial attacks. arXiv preprint arXiv:1811.06492, 2018.

[14] W. E. A proposal on machine learning via dynamical systems. Communications in Mathematics and Statistics, 5:1–11, 2017.

[15] M. Giles. Multilevel monte carlo methods. Acta Numerica, pages 1–70, 2018.

[16] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6275, 2014.

[17] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel. Adversarial perturbations against deep neural networks for malware classification. arXiv preprint arXiv:1606.04435, 2016.

[18] A. Guisti, J. Guzzi, D.C. Ciresan, F.L. He, J.P. Rodriguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di Carlo, and et al. A machine learning approach to visual perception of forecast trails for mobile robots. IEEE Robotics and Automation Letters, pages 661–667, 2016.

[19] E. Haber and L. Ruthotto. Stable architectures for deep neural networks. Inverse Problems, 34:014004, 2017.

[20] K. L. Hansen and P. Salamon. Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence archive, pages 993–1001, 1990.

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, pages 770–778, 2016.

[22] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In ECCV, 2016.

[23] G. Huang, Z. Liu, L. van der Maaten, and K. Weinberger. Densely connected convolutional networks. In CVPR, 2017.

[24] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin. Black-box adversarial attacks with limited queries and information. International Conference on Machine Learning, 2018.

[25] C. Ju, A. Bibaut, and M. J. van der Laan. The relative performance of ensemble methods with deep convolutional neural networks or image classification. arXiv preprint arXiv:1607.02533, 2016.

[26] M. Kac. On distributions of certain Wiener functionals. Transactions of the American Mathematical Society, 65:1–13, 1949.

[27] D. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[28] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, 2012.

[30] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. In International Conference on Learning Representations, 2017.

[31] O. Ladyzhenskaja, V. Solonnikov, and N. Uraltseva. Linear and Quasilinear Equations of Parabolic Type. American Mathematical Society, Providence, R.I., 1968.

[32] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. Nature, 521:436–444, 2015.

[33] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana. Certified robustness to adversarial examples with differential privacy. In IEEE Symposium on Security and Privacy (SP), 2019.

[34] B. Li, C. Chen, W. Wang, and L. Carin. Second-order adversarial attack and certifiable robustness. arXiv preprint arXiv:1809.03113, 2018.

[35] Z. Li and Z. Shi. Deep residual learning and pdes on manifold. arXiv preprint arXiv:1708:05115, 2017.

[36] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks. arXiv preprint arXiv:1611.02770, 2016.

[37] Y. Lu, A. Zhong, Q. Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. International Conference on Machine Learning, 2018.

[38] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In International Conference on Learning Representations, 2018.

[39] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[40] Taesik Na, Jong Hwan Ko, and Saibal Mukhopadhyay. Cascade adversarial machine learning regularized with a unified embedding. In International Conference on Learning Representations, 2018.

[41] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. IEEE European Symposium on Security and Privacy, pages 372–387, 2016.

[42] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman. Sok: Towards the science of security and privacy in machien learning. arXiv preprint arXiv:1611.03814, 2016.

[43] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. CoRR, abs/1605.07277, 2016.

[44] A. Raghunathan, J. Steinhardt, and P. Liang. Certified defenses against adversarial examples. In International Conference on Learning Representations, 2018.

[45] A. Raghunathan, J. Steinhardt, and P. Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In Advances in Neural Information Processing Systems, 2018.

[46] A. Ross and F. Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. arXiv preprint arXiv:1711.09404, 2017.

[47] H. Salman, G. Yang, H. Zhang, C. Hsieh, and P. Zhang. A convex relaxation barrier to tight robustness verification of neural networks. arXiv preprint arXiv:1902.08722, 2019.

[48] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[49] T. Strauss, M. Hanselmann, A. Junginger, and H. Ulmer. Ensemble methods as a defense to adversarial perturbations against deep neural networks. arXiv preprint arXiv:1709.0342, 2017.

[50] Q. Sun, Y. Tao, and Q. Du. Stochastic training of residual networks: a differential equation viewpoint. arXiv preprint arXiv:1812.00174, 2018.

[51] 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.

[52] B. Wang, A. T. Lin, Z. Shi, W. Zhu, P. Yin, A. L. Bertozzi, and S. J. Osher. Adversarial defense via data dependent activation function and total variation minimization. arXiv preprint arXiv:1809.08516, 2018.

[53] B. Wang, X. Luo, Z. Li, W. Zhu, Z. Shi, and S. Osher. Deep neural nets with interpolating function as output activation. Advances in Neural Information Processing Systems, 2018.

[54] E. Wong and J. Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In International Conference on Machine Learning, 2018.

[55] E. Wong, F. Schmidt, J. Metzen, and J. Kolter. Scaling provable adversarial defenses. In Advances in Neural Information Processing Systems, 2018.

[56] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In CVPR, 2017.

[57] D. Yin, K. Ramchandran, and P. Bartlett. Rademacher complexity for adversarially robust generalization. arXiv preprint arXiv:1810.11914, 2018.

[58] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In BMVC, 2016.

[59] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. Ghaoui, and M. Jordan. Theoretically principled trade-off between robustness and accuracy. arXiv preprint arXiv:1901.08573, 2019.

[60] S. Zheng, Y. Song, T. Leung, and I. Goodfellow. Improving the robustness of deep neural networks via stability training. In IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[61] M. Zhu, B. Chang, and C. Fu. Convolutional neural networks combined with runge-kutta methods. arXiv preprint arXiv:1802.08831, 2018.