CHAPTER 1

Background and motivation

1.1 What is adversarial machine learning?

Adversarial machine learning (AdvML) refers to the methodology of introducing a virtual adversary for evaluating and improving the performance of a machine learning (ML) system throughout its lifecycle of development and deployment, ranging from training (e.g., data collection, model selection and tuning, etc), model testing (e.g., vulnerability assessment, performance benchmarking, etc), hardware implementation, and system integration to continuous system status monitoring and updates.

We list two primary scientific and engineering goals considered in AdvML:

1. The practice of virtual adversary for proactive risk evaluation to prevent or mitigate different kinds of failure modes of machine learning systems when deployed in the real world. The failure modes include natural changes such as domain shifts in data inputs and lacking generalization to unseen or out-of-domain data, as well as potential adversarial threats (from real adversary) such as training-phase and deployment-phase attacks aiming to compromising machine learning algorithms and systems.
2. The use of virtual adversary to deliver new machine learning algorithms for performance improvement. Comparing to standard ML without the notion of adversary, the interplay between the model of interest and virtual adversary, either in cooperative or competitive manner, can help developing more effective and robust machine learning models and algorithms. One well-known example is the training of generative adversarial network (GAN) (Goodfellow et al., 2014), which attains a high-quality generator via introducing a discriminator.

Despite the fact that the second goal touches upon many related research topics associated with AdvML, such as GANs, multiagent systems, and game-oriented learning, this book focuses on the topics underlying adversarial robustness in machine learning algorithms and systems. These topics cover both aforementioned goals and deliver important insights to ML applications concerning safety, security, and reliability.

Adversarial Robustness for Machine Learning
https://doi.org/10.1016/B978-0-12-824020-5.00009-0

Table 1.1 Commonly used mathematical symbols and their meanings.

Symbol	Meaning
$(\boldsymbol{x}, \boldsymbol{y}) \sim \mathcal{D}$	data sample $\boldsymbol{x}$ and one-hot coded label $\boldsymbol{y}$ drawn from the data distribution $\mathcal{D}$
K	number of classes in a classification task
$\boldsymbol{\theta}$	parameters of a machine learning model
$f_{\boldsymbol{\theta}} : \mathbb{R}^d \mapsto [0, 1]^K$ (or $F_{\boldsymbol{\theta}}$)	K-way classifier (e.g., a neural network parameterized by $\boldsymbol{\theta}$)
$\boldsymbol{\delta}$ (or $\boldsymbol{\Delta}$)	input perturbation
ϵ	perturbation budget
$\nabla_{\boldsymbol{z}} J(\cdot)$	gradient of a scalar function J with respect to $\boldsymbol{z}$
$\mathcal{L}$ (or loss, J)	loss function
$[K]$	integer set $\{1, 2, \ldots, K\}$, where $K \geq 1$.
$\|\boldsymbol{x}\|_p$ $(p \geq 1)$	vector p-norm of $\boldsymbol{x} = [x_1, \ldots, x_d]$, defined as $\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^d |x_i|^p\right)^{1/p}$
$\|\boldsymbol{x}\|_\infty$	infinity norm of $\boldsymbol{x} = [x_1, \ldots, x_d]$, defined as $\|\boldsymbol{x}\|_\infty = \max_{i \in [d]} \|x_i\|$
$\text{sign}(\cdot)$	elementwise sign operation; $\text{sign}(z) = 1$ if $z \geq 0$ and $\text{sign}(z) = -1$ otherwise.

1.2 Mathematical notations

Throughout this book, unless specified otherwise, we will use the following convention for mathematical notations. Regular letters (e.g., x or X) are used to denote scalars, index, or an element in a set. Bold-faced lowercase letters (e.g., $\boldsymbol{x}$) are used to denote column vectors. Bold-faced uppercase letters (e.g., $\boldsymbol{X}$) are used to denote matrices. Uppercase letters in calligraphic fonts (e.g., $\mathcal{X}$) are used to denote sets or probability distributions. All vectors and matrices are assumed to be real-valued. The subscript x_i (X_{ij}) denotes the ith element (the element at the ith row and the jth column) of a vector $\boldsymbol{x}$ (a matrix $\boldsymbol{X}$). $[\boldsymbol{x}]_j$ (or x_j) means the jth element of the vector $\boldsymbol{x}$. The notation $\mathbb{R}^d$ denotes the space of d-dimensional real-valued vectors. The notation $\cdot^T$ denotes the transpose of a vector or a matrix.

Table 1.1 summarizes commonly used mathematical symbols and their meanings in this book. In each chapter the associated mathematical notations and their meanings will be formally defined. Depending on the context, the notation $f_{\boldsymbol{\theta}}(\cdot)$ (or $F_{\boldsymbol{\theta}}(\cdot)$) for model output may refer to the top-1 (mostly likely) class based on the model prediction. Similarly, the notation y may be used to denote the groundtruth class label of a data sample $\boldsymbol{x}$.

1.3 Machine learning basics

Machine learning is the methodology of teaching machines for problem/task solving based on the observable data (or the interaction with training environments) and the underlying computational/statistical learning mechanism. The learning component, also known as model training, involves a parameterized model whose parameters (or model weights) are updated by a designated loss function, measured by the available data or rewards from the training environment. The updates of the model parameters are often accomplished by gradient-based optimization algorithms such as stochastic gradient descent. Here "stochastic" means that in each iteration of the optimization process, a subset of data samples are sampled from the whole training dataset (i.e., a minibatch) for evaluating the loss and calculating the gradient with respect to the model parameters for updates. The science and engineering of machine learning have been a mainstream research field and dominant technology in artificial intelligence and computer science, with far-reaching applications to specific domains such as computer vision, natural language processing, policy learning, robot planning, speech processing, healthcare, data science, to name a few.

Supervised machine learning is a major branch of machine learning, which uses a set of labeled training data samples $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^n$ and a designated loss function $\mathcal{L}$ for training the parameters $\boldsymbol{\theta}$ associated with a machine learning model $f_{\boldsymbol{\theta}}$. Most of the supervised learning methods follow the Empirical Risk Minimization (ERM) framework, where the model parameters $\boldsymbol{\theta}$ is obtained by solving the following optimization problem:

$$\min_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i),$$

where the loss function $\mathcal{L}(\cdot, \cdot)$ measures how the model's prediction fits the label. This optimization problem can be typically solved by Stochastic Gradient Descent (SGD) or other gradient-based optimizers such as Adagrad (Duchi et al., 2011) and Adam (Kingma and Ba, 2015). The loss function can be designed according to the application. Popular loss functions include the mean squared error (MSE) loss defined as $\mathcal{L}_{\text{MSE}}(f_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{y}) = \|f_{\boldsymbol{\theta}}(\boldsymbol{x}) - \boldsymbol{y}\|_2^2$, and the cross entropy (CE) loss defined as $\mathcal{L}_{\text{CE}}(f_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{y}) = -\sum_{k=1}^{K} [\boldsymbol{y}]_k \log [f_{\boldsymbol{\theta}}(\boldsymbol{x})]_k$, where in the CE loss $f_{\boldsymbol{\theta}}$ outputs a vector on the K-dimensional probability simplex such that $f_{\boldsymbol{\theta}}(\cdot) \in [0, 1]^K$ and $\sum_{k=1}^{K} [f_{\boldsymbol{\theta}}(\cdot)]_k = 1$.

Semisupervised machine learning refers to the problem setting of leveraging a labeled dataset (usually limited) and an unlabeled dataset (usually abundant) for machine learning. Unsupervised machine learning means learning representations of the data inputs without using any data labels. In particular, self-supervised machine learning uses self-generated pseudo-labels or tasks for learning good representations. For unsupervised or self-supervised machine learning schemes, the practice is to follow the strategy of pretraining (on a large unlabeled dataset) and fine-tuning (on a task-specific labeled dataset). Transfer learning refers to fine-tuning a task-specific model trained in a source domain to solve a related task in a target domain using the labeled target-domain data.

Scope of this book. Most of the considered machine learning tasks and models in this book are within the scope of supervised machine learning, such that the goal of the adversary can be defined in a straightforward manner. Nonetheless, the methodology can be extended to unsupervised or self-supervised machine learning settings, as discussed in Chapter 21. In this book, the studied machine learning model, task, and loss function will be introduced and clearly defined in each chapter; we defer the readers interested in the in-depth background of machine learning to seminal books such as (Bishop, 2006).

Neural networks. A neural network is the default machine learning model in deep learning (LeCun et al., 2015), a powerful and high-capacity tool for representation learning. Deep neural networks (DNNs) have achieved state-of-the-art performance in many machine learning tasks. In general, neural networks perform a set of layered operations on data inputs, in either sequential or recurrent manner, with layers of trainable parameters (mostly linear or convolutional transformations), nonlinear activation functions, or dimension reduction through max/average pooling. The fact that neural networks can be "deep" means that a large number of such layers can constitute a high-capacity machine learning model with a strong expressive power of data representations and function approximations. With sufficient training data and compute power, neural networks are capable of capturing the complex relationship between data samples and associated labels, as well as learning generalizable representations for different data modalities. In this book, we will primary focus on the adversarial robustness of neural networks because they are state-of-the-art machine learning models. Nonetheless, the methodology naturally applies to other machine learning models such as support vector machines, random forests, gradient-boosted trees, etc. In fact, the black-box adversarial attacks introduced in Chapter 3

are agnostic to the underlying machine learning model used for classification. The necessary details of the studied neural networks will be given in the corresponding chapters.

Commonly used datasets. Below we summarize some commonly used datasets for image classification.

- MNIST (LeCun et al., 1998): The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits (0 to 9). It has a training set of 60,000 examples and a test set of 10,000 examples.
- CIFAR-10 (Krizhevsky et al., 2009): CIFAR-10 data samples are labeled subsets of the 80 million tiny images dataset. The dataset consists of 60,000 32×32 color images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images.
- ImageNet (1K) (Deng et al., 2009): ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of image. We refer the ImageNet dataset to the Large Scale Visual Recognition Challenge 2012 (ILSVRC2012), which is a subset of the large hand-labeled ImageNet dataset (10,000,000 labeled images depicting 10,000+ object categories). The training data is a subset of ImageNet containing the 1,000 categories and about 1.2 million images. The test data has 50,000 images.

1.4 Motivating examples

This section provides two motivating examples to highlight the importance of studying adversarial robustness for machine learning.

Adversarial robustness $\neq$ accuracy – what standard accuracy fails to tell

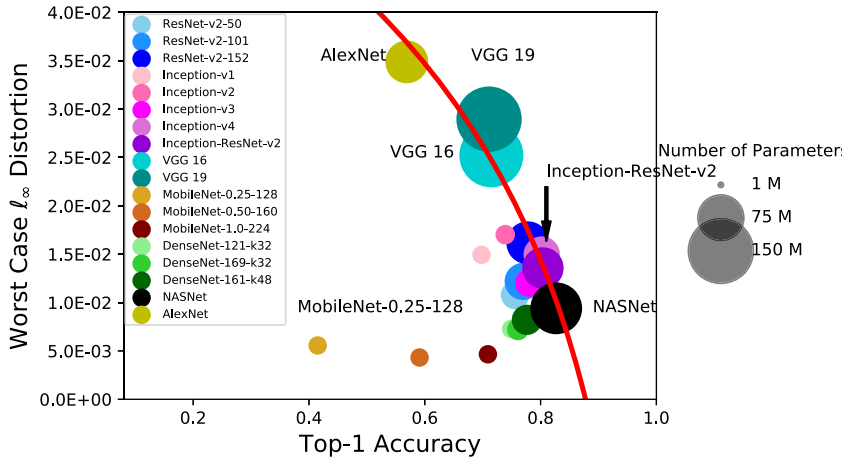
The prediction accuracy has been the long-lasting and sole standard for comparing the performance of different image classification models, including the yearly ImageNet competition (Russakovsky et al., 2015) held from 2010 to 2017. Many significant advances in the architecture design and training of neural networks can be attributed to this task. In the competition a model with a higher standard accuracy (e.g., top-1 or top-5 prediction accuracy on the test dataset) implies that a model is "better". However, surprisingly, (Su et al., 2018) performed a large-scale adversarial robustness study on 18 different publicly available ImageNet models and

discovered that the empirical ℓ_2 and ℓ_∞ distortion metrics scale linearly with the logarithm of classification error. Their results suggest that when using the standard accuracy as the sole metric to benchmark, the model performance may give a false sense of progress in machine learning, because the models having higher standard accuracy (lower classification error) are also shown to be more sensitive to input perturbation leading to prediction changes.

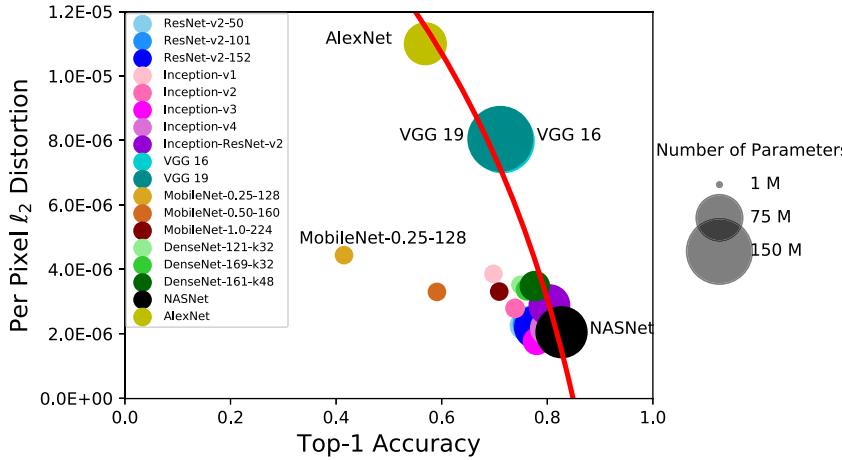
For each of the 18 models, Su et al. (2018) apply different attacks to generate adversarial examples for a common set of originally correctly classified data samples to find out the smallest additive distortions (measured by ℓ_p norms) required to flip the model prediction. They study the empirical relation between adversarial robustness and (standard) accuracy of different ImageNet models, where the robustness is evaluated in terms of the minimum ℓ_∞ and ℓ_2 distortion metrics from successful I-FGSM (Kurakin et al., 2016) and C&W (Carlini and Wagner, 2017b) attacks, respectively.

The scatter plots of distortions v.s. top-1 prediction accuracy are displayed in Fig. 1.1. We define the classification error as 1 minus top-1 accuracy (denoted as 1-acc). By regressing the distortion metric with respect to the classification error of networks on the Pareto frontier of robustness-accuracy distribution (i.e., AlexNet, VGG 16, VGG 19, ResNet_v2_152, Inception_ResNet_v2, and NASNet), they find that the distortion scales linearly with the logarithm of classification error. That is, the distortion and classification error has the following relation: $\text{distortion} = a + b \cdot \log(\text{classification-error})$. The fitted parameters of a and b are given in the captions of Fig. 1.1. Taking I-FGSM attack (Kurakin et al., 2016) as an example, the linear scaling law suggests that to reduce the classification error by a half, the ℓ_∞ distortion of the resulting network will be expected to reduce by approximately 0.02, which is roughly 60% of the AlexNet distortion. Following this trend, if we naively pursue a model with low test error, then the model's adversarial robustness may suffer. Consequently, when designing new networks for ImageNet, standard accuracy is not sufficient to characterize the model performance against adversarial attacks. Similar trend is observed by Su et al. (2018) when using an attack-agnostic adversarial robustness metric (the CLEVER score (Weng et al., 2018b)) as the y-axis.

This undesirable trade-off between standard accuracy and adversarial robustness suggests that one should employ the techniques discussed in this book to evaluate and improve adversarial robustness for machine learning.



(a) Fitted Pareto frontier of ℓ_∞ distortion (I-FGSM attack) vs. top-1 accuracy:
ℓ_∞ dist = $[2.9 \cdot \ln(1\text{-acc}) + 6.2] \times 10^{-2}$



(b) Fitted Pareto frontier of ℓ_2 distortion (C&W attack) vs. top-1 accuracy:
ℓ_2 dist = $[1.1 \cdot \ln(1\text{-acc}) + 2.1] \times 10^{-5}$

Figure 1.1 Robustness vs. classification accuracy plots of I-FGSM attack (Kurakin et al., 2016), C&W attack (Carlini and Wagner, 2017b), and on random targets over 18 ImageNet models (Su et al., 2018).

Fast adaptation of adversarial robustness evaluation assets for emerging machine learning models

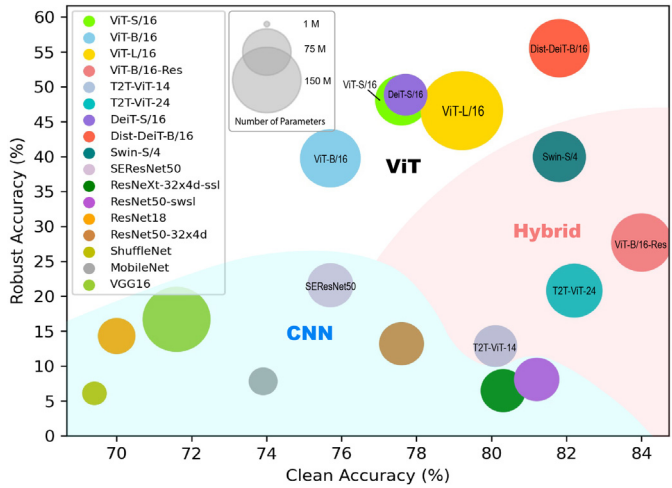
As another motivating example, once we have sufficient practice in studying adversarial robustness, when new machine learning models emerge, we can

quickly adapt the existing adversarial robustness tools for evaluation and profiling.

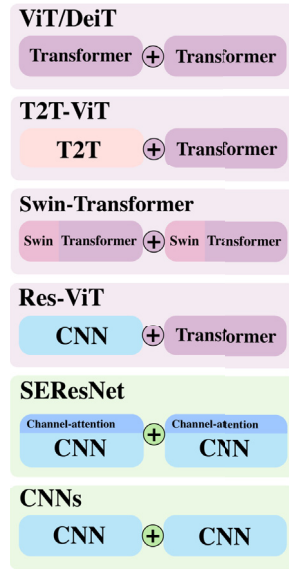
For instance, transformers are originally applied in natural language processing (NLP) tasks as a type of deep neural network (DNN) mainly based on the self-attention mechanism (Vaswani et al., 2017; Devlin et al., 2018; Brown et al., 2020b), and transformers with large-scale pretraining have achieved state-of-the-art results on many NLP tasks (Devlin et al., 2018; Liu et al., 2019e; Yang et al., 2019b; Sun et al., 2019b). Recently, Dosovitskiy et al. (2020) applied a pure transformer directly to sequences of image patches (i.e., a vision transformer, ViT) and showed that the Transformer itself can be competitive with convolutional neural networks (CNNs) on image classification tasks. Since then, transformers have been extended to various vision tasks and show competitive or even better performance compared to CNNs and recurrent neural networks (RNNs) (Carion et al., 2020; Chen et al., 2020b; Zhu et al., 2020b).

Using existing tools for adversarial robustness evaluation, Shao et al. (2021a) examine the adversarial robustness of ViTs on image classification tasks and make comparisons with CNN baselines. As highlighted in Fig. 1.2, their experimental results illustrate the superior robustness of ViTs than CNNs in both white-box and black-box attack settings, based on which they make the following important findings:

- Features learned by ViTs contain less low-level information and benefit adversarial robustness. ViTs achieve a lower attack success rate (ASR) of 51.9% compared with a minimum of 83.3% by CNNs in Fig. 1.2. They are also less sensitive to high-frequency adversarial perturbations.
- Using denoised randomized smoothing (Salman et al., 2020b), ViTs attain significantly better certified robustness than CNNs.
- It takes the cost of adversarial robustness to improve the classification accuracy of ViTs by introducing blocks to help learn low-level features as shown in Fig. 1.2.
- Increasing the proportion of transformer blocks in the model leads to better robustness when the model consists of both transformer and CNN blocks. For example, the attack success rate (ASR) decreases from 87.1% to 79.2% when 10 additional transformer blocks are added to T2T-ViT-14. However, increasing the size of a pure transformer model cannot guarantee a similar effect, e.g., the robustness of ViT-S/16 is better than that of ViT-B/16 in Fig. 1.2.
- Pretraining without adversarial training on larger datasets does not improve adversarial robustness though it is critical for training ViT.



(a) Robust Accuracy v.s. Clean Accuracy



(b) Model Architecture

Figure 1.2 (a) Robust accuracy v.s. clean accuracy. The robust accuracy is evaluated by AutoAttack (Croce and Hein, 2020). The "Hybrid" class includes CNN-ViT, T2T-ViT and Swin-T as introduced by Shao et al. (2021a). Models with attention mechanisms have their names printed at the center of the circles. ViTs have the best robustness against adversarial perturbations. Introducing other modules to ViT can improve clean accuracy but hurt adversarial robustness. CNNs are more vulnerable to adversarial attacks. (b) Illustration of considered vision transformers and CNNs in Shao et al. (2021a).

- The principle of adversarial training through min-max optimization (Madry et al., 2017; Zhang et al., 2019b) can be applied to train robust ViTs.

Paul and Chen (2022) conduct a comprehensive study on the general robustness of ViT against common corruptions, adversarial perturbations, distribution shifts, and natural adversarial examples. They use six different diverse ImageNet datasets concerning robust classification to conduct a comprehensive performance comparison of ViT models and state-of-the-art CNNs. Through a series of six systematically designed experiments, they show both quantitative and qualitative indications to explain why ViTs are indeed more robust learners.

These important results and insights into adversarial robustness could not be obtained shortly after the emergence of new machine learning models such as ViTs without reusing the available assets and tools in studying existing machine learning models. More importantly, as long as we acknowledge that ensuring adversarial robustness is a necessary step for making realistic and sustainable improvement in machine learning, stronger momentum will be generated to accelerate the progress in developing advanced machine learning.

1.5 Practical examples of AI vulnerabilities

Here are some public resources summarizing real-world incidences and vulnerabilities in an ML-based system.

- Adversarial ML Threat Matrix: https://github.com/mitre/advmlthreatmatrix.
- AI Incidence Database: https://incidentdatabase.ai/.

1.6 Open-source Python libraries for adversarial robustness

Here are some resources of open-source Python-based libraries for studying adversarial robustness:

- IBM adversarial robustness toolbox: https://github.com/Trusted-AI/adversarial-robustness-toolbox.
- CleverHans: https://github.com/cleverhans-lab/cleverhans.
- Foolbox: https://github.com/bethgelab/foolbox.
- RobustBench: https://robustbench.github.io/.
- TextAttack: https://github.com/QData/TextAttack.