Doctoral Qualifying Examination

Ramesh Kumar Sah

Washington State University

March 25, 2020

Table of Contents

Adversarial Machine Learning

Adversarial Examples

Definition

"*Adversarial examples are inputs formed by applying small but intentional perturbation to the inputs from the dataset such that the perturbed inputs are almost indistinguishable from the true inputs and results in the model outputting an incorrect answer with high confidence*" [1, p. 1].

Mathematically for model M trained on samples (x, y) from distribution D, an adversarial example is defined as

$$\bar{x} = x + \delta \tag{1}$$

such that for a metric m and a constant ϵ

$$\begin{aligned} M(\bar{x}) &\neq M(x) \\ m(\bar{x} - x) &\leq \epsilon \end{aligned} \tag{2}$$

Here, δ is the adversarial perturbation computed using adversarial attack methods. One of the norms l_0, l_2 or l_∞ is used as the metric m.

Some Adversarial Examples

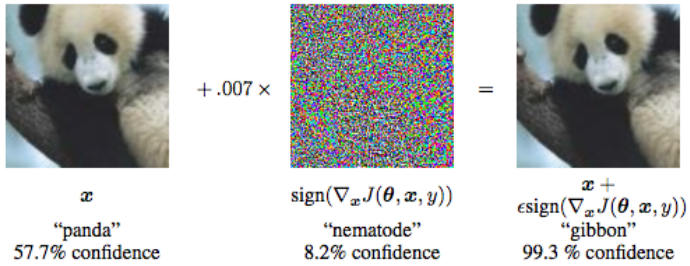


Figure: Image Adversarial Example. Source [1].

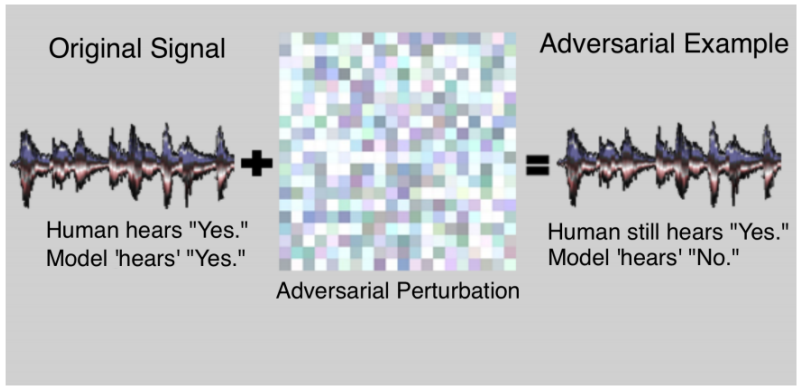


Figure: Audio Adversarial Example. Source [3].

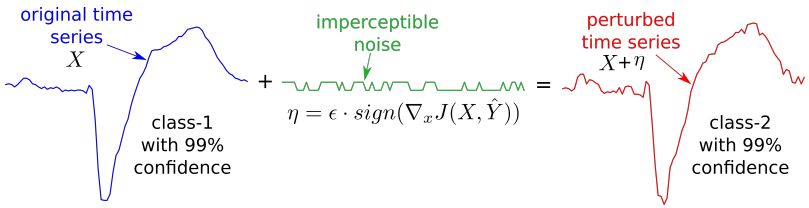


Figure: Time-Series Adversarial Example. Source [2].

Adversarial Attacks

Adversarial attacks can be divided into three categories.

1. Poisoning Attack
2. Evasion Attack
3. Exploratory Attack

Poisoning Attack

- Adversary wants to make the model M learn false connection between inputs x and outputs y by corrupting the training data.
- In this mode, the adversary wants to influence the learning of the model M to make it do its bidding, without needing to alter the inputs at test time.

Evasion Attack

- In evasion attack the adversary modifies the inputs x to fool the model M.
- Evasion attack is further sub-divided into two types:
  1. Untargeted Attack: In untargted attack the adversary wants to cause random misclassification. That is for an adversarial example $\bar{x}$, we have $M(\bar{x}) \neq M(x)$.
  2. Targeted Attack: In targeted attack the adversary chooses the target class y_{tar} in which it wants the model M classify the adversarial example $\bar{x}$, i.e., $M(\bar{x}) = y_{tar}$.

Exploratory Attack

- In this setting, the adversary wants to gain as much knowledge as possible about the target system.
- Exploratory attacks are carried out when the adversary does not have access to the target system.

Adversary Knowledge

- The difficulty in attack a target system depends on the type of attack an adversary wants to mount and the knowledge of the target system the adversary has.
- Depending on the adversary's knowledge of the target system it will operate in one of the three possible operating settings.
  1. White-box Setting
  2. Gray-box Setting
  3. Black-box Setting

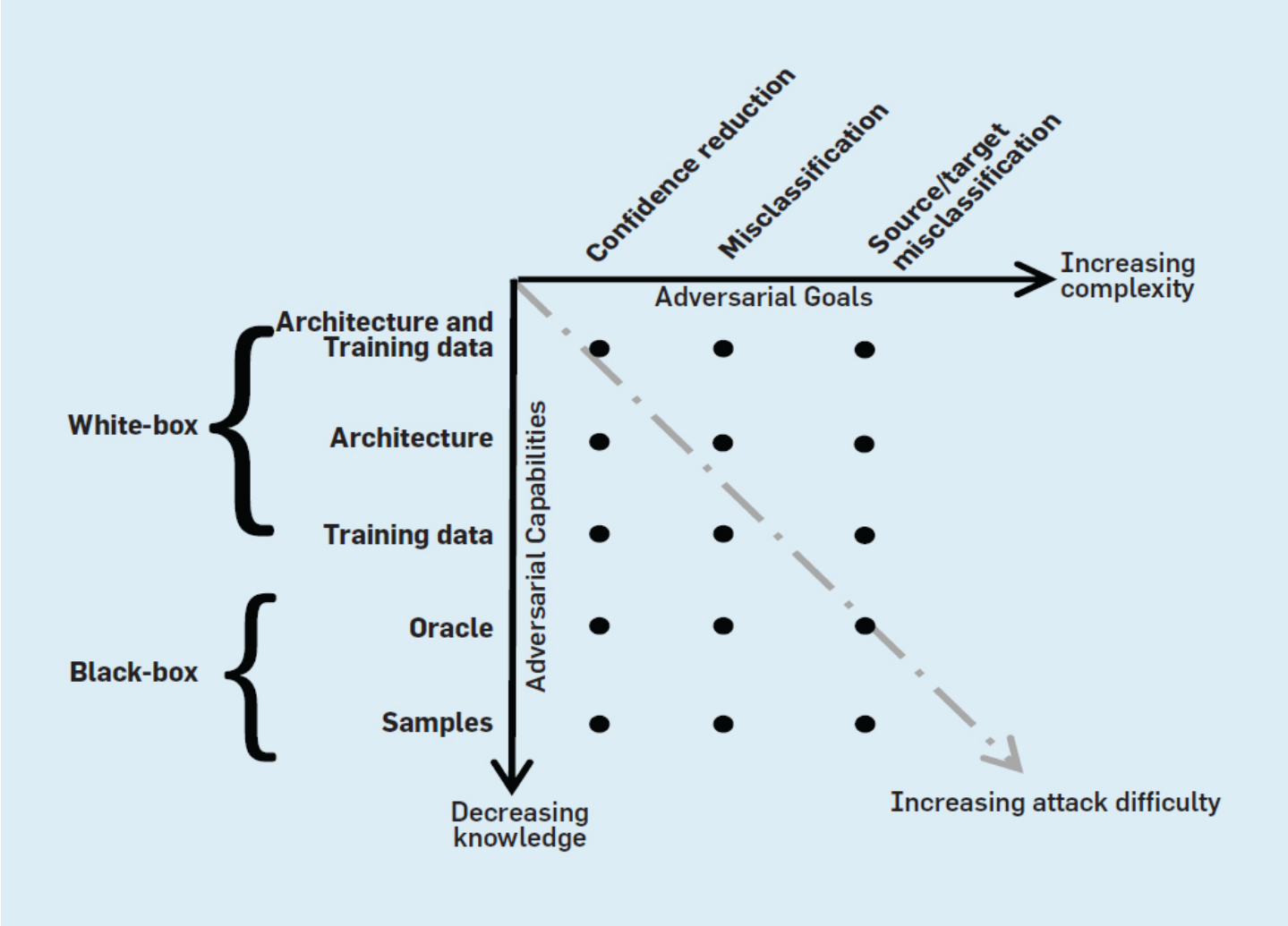


Figure: The x-axis represents adversarial goals in terms of different attack methods, and the y-axis shows the knowledge of the adversary. Source [4]

Adversarial Transferability

Definition

Adversarial examples computed for a model is often also effective against other independently trained models. This is called adversarial transferability.

- Transferability of adversarial examples makes real-world black-box attacks possible.
- An adversary can compute adversarial perturbation for each inputs separately (instance-specific) or for all inputs at once (instance-agnostic).
- Instance-agnostic perturbations are called universal adversarial perturbation.

Adversarial Examples Are Not Bugs, They Are Features

Introduction

- In this work [5], the authors have presented a novel theory for the existence of adversarial examples. They prove their arguments with extensive set of experiments.
- They separate robust and non-robust features present in the data to train different models and do validations.
- Their main results is “adversarial examples are possible due to existence of non-robust features in the dataset”.

Contributions

The authors have made the following contributions in [5]:

1. Presented a new theory that explains the reasons for the existence of adversarial examples. They have based their theory on the data rather than the model and learning algorithm.
2. Linked adversarial transferability with their theory, attributing adversarial transferability to the presence of non-robust features that are learned my multiple models.
3. Provided explanation for some existing results and refuted theories that attributed adversarial examples to high dimensionality of input space and finite-sample overfitting.

A Simple Experiment

Consider an image of a dog taken from CIFAR-10 dataset D,

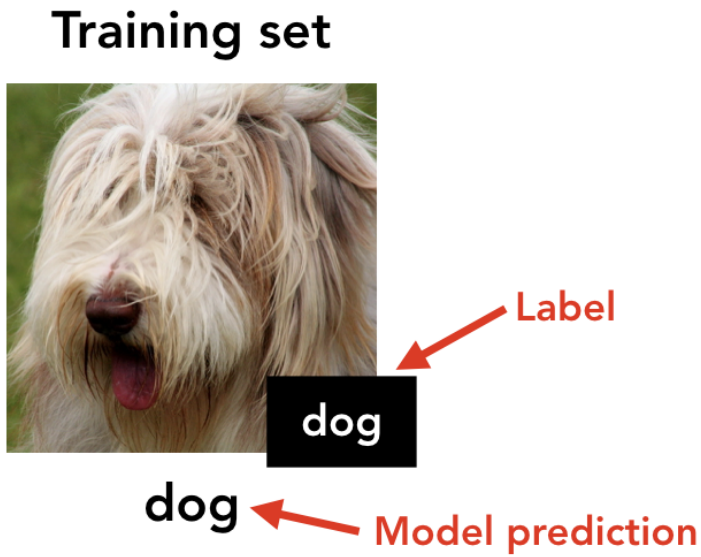


Figure: An image of dog taken from CIFAR-10 dataset. Source [5]

We compute targeted adversarial examples for all $(x, y) \in D$ to create a new dataset $\hat{D}$. For example

Figure: An adversarial image of dog classified and labeled as cat. Source [5]

- The dataset $\hat{D}$ looks completely mislabeled to human because humans rely solely on robust features for classification.
- A new classifier $\hat{M}$ trained on $\hat{D}$ showed moderate accuracy of 45% on the original test set from D [5].

This means that

1. training inputs in $\hat{D}$ are associated with their true label (present in D) solely through imperceptible adversarial perturbations and are associated with the incorrect target label through all visible features
2. adversarial perturbations of standard model (M trained on D) are patterns predictive of the target class in a well-generalizing sense.
3. there exist a variety of features of the input that are predictive of the label, and only some of these are perceptible to humans.

Robust and Non-Robust Features

Predictive features of the data can be split into robust and non-robust features.

- Robust features: patterns that are predictive of the true label even when adversarially perturbed.
- Non-robust features: patterns that are predictive, but can be flipped by an adversary to indicate wrong class.

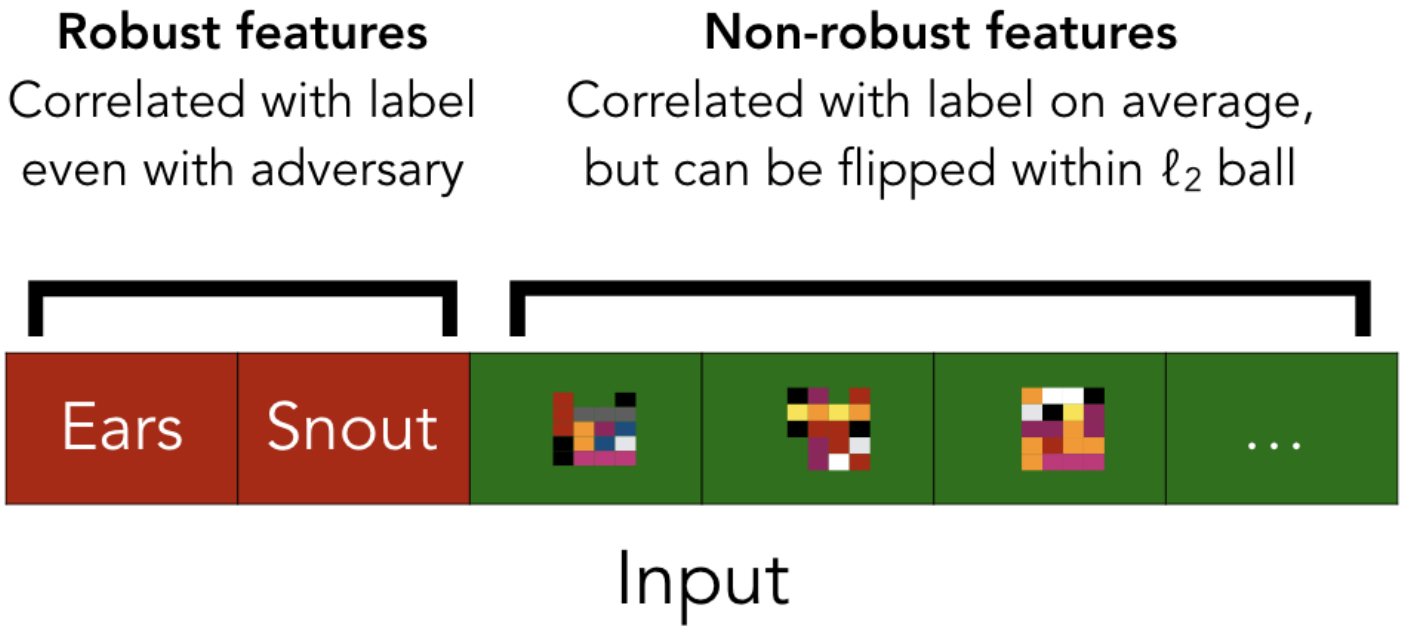


Figure: The representation of robust and non-robust features. Source [5]

In the original dataset D, robust and non-robust features are associated and predictive of the true class, in this case "dog".

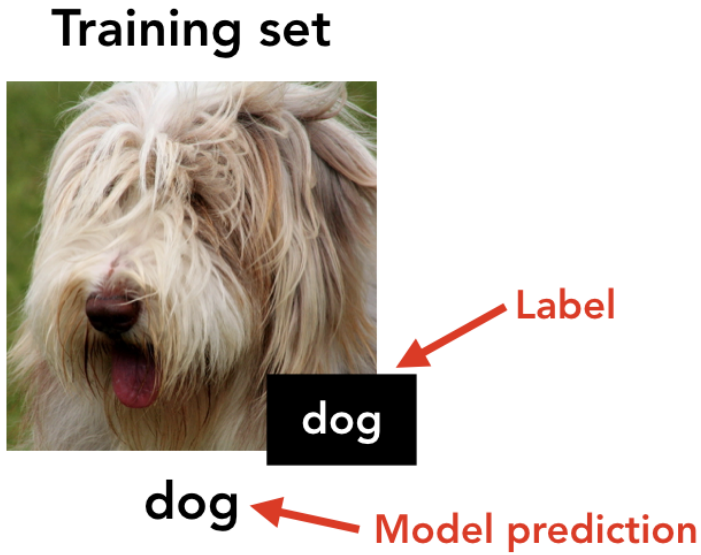


Figure: An image of dog taken from CIFAR-10 dataset. Source [5]

In the adversarial dataset $\hat{D}$, robust features are wrongly associated with the adversarial target class "cat" and the non-robust features that were modified by the adversary are predictive of the class "cat" in general.

Figure: An adversarial image of dog classified and labeled as cat. Source [5]

And, deep neural networks rely on non-robust features, even in the presence of robust features.

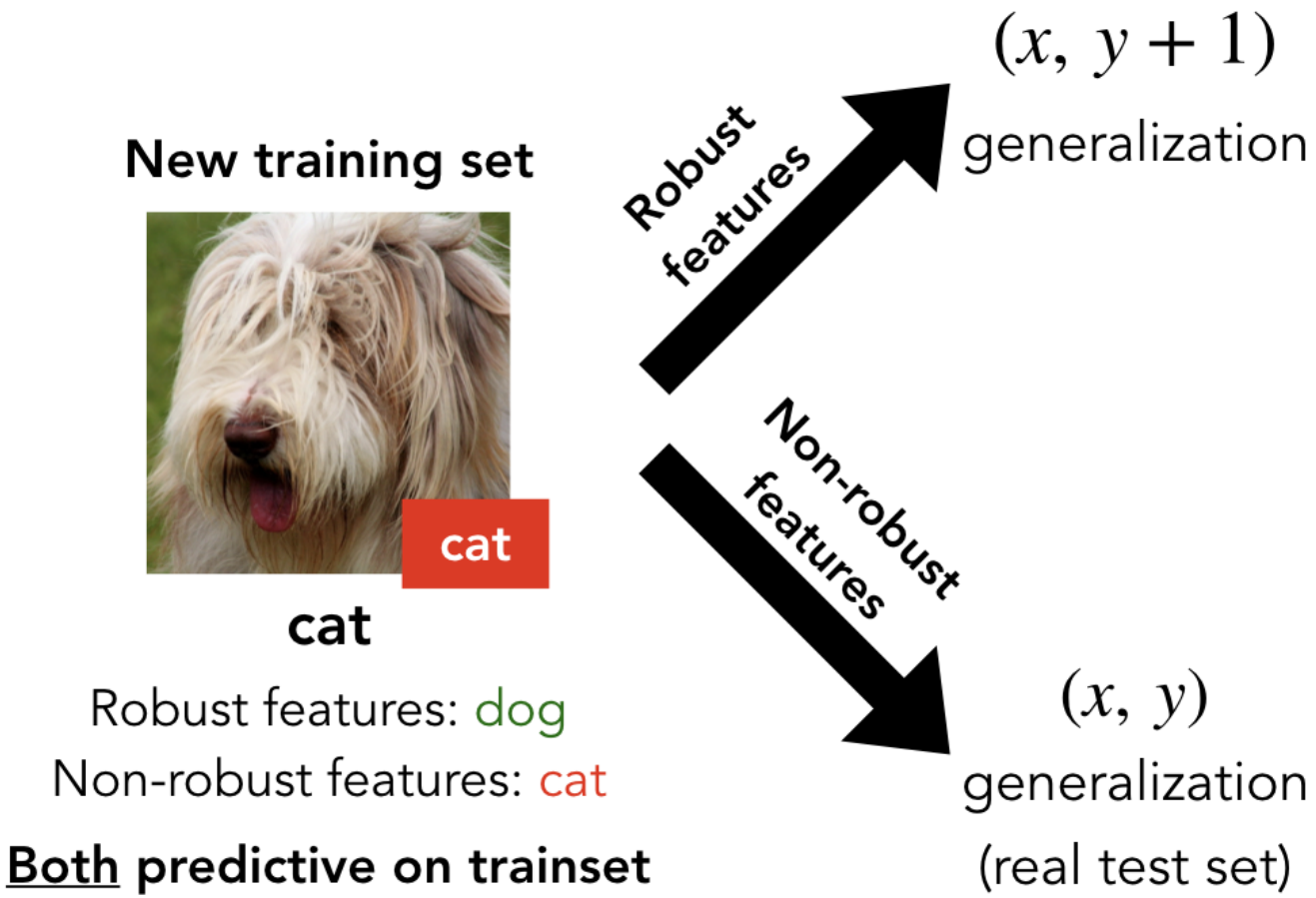


Figure: The complete picture of robust and non-robust features. Source [5]

Datasets

Next generate robust $\hat{D}_R$ and non-robust dataset $\hat{D}_{NR}$ from D. And train different machine learning models using standard loss and adversarial loss.

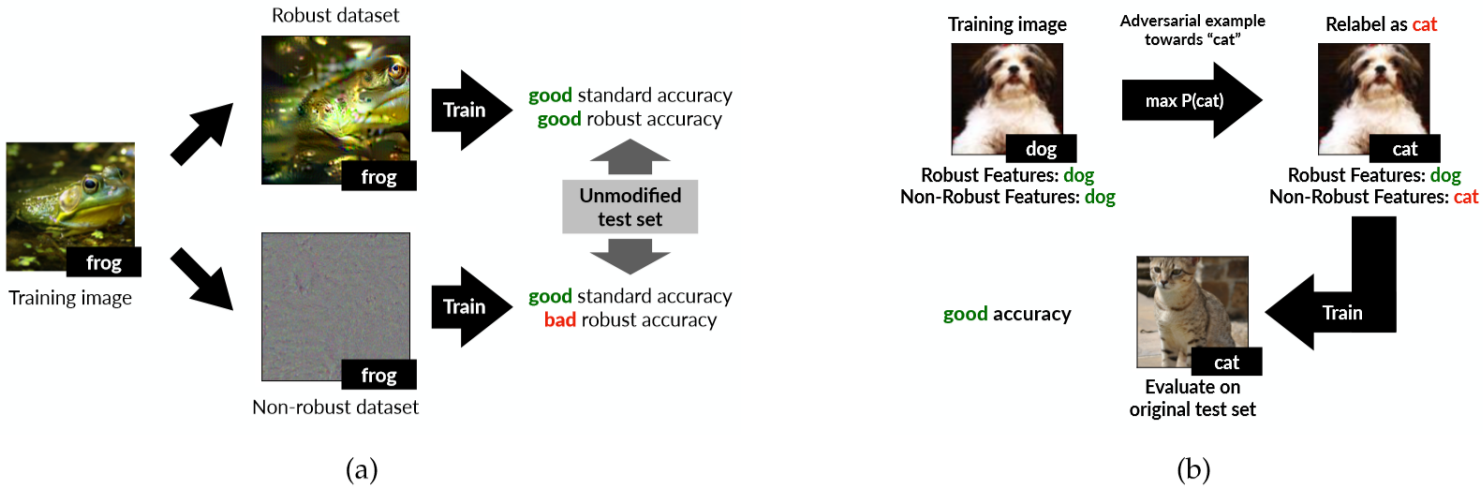


Figure: Generating robust and non-robust datasets. Source [5]

Results

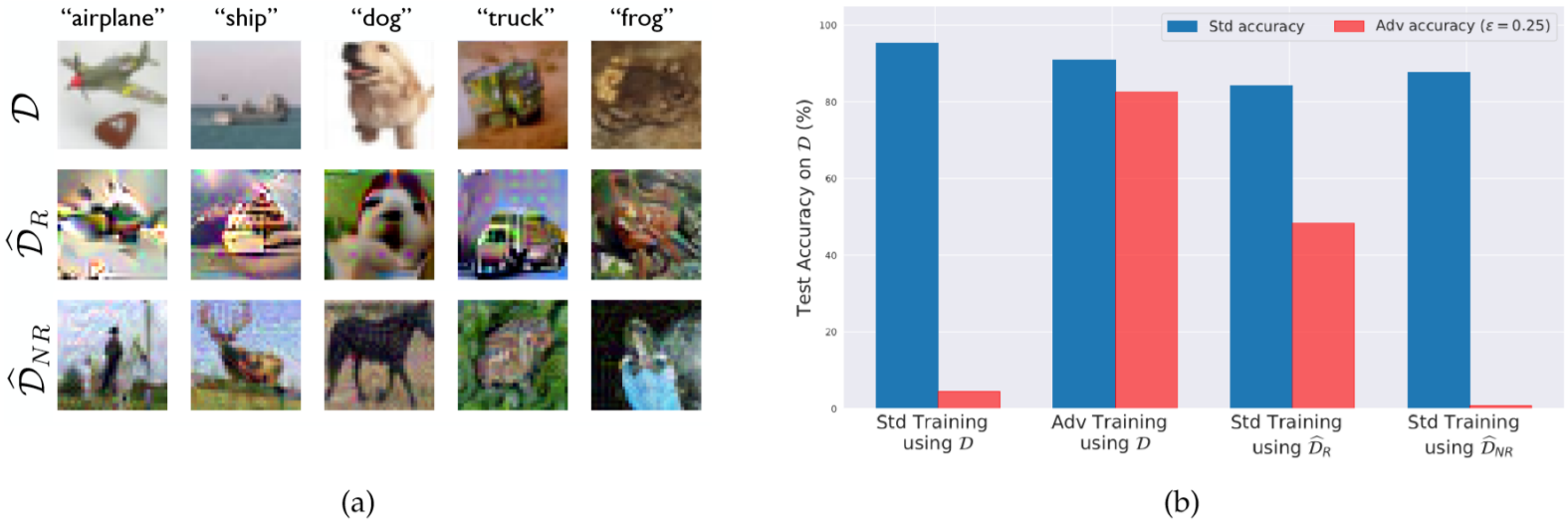
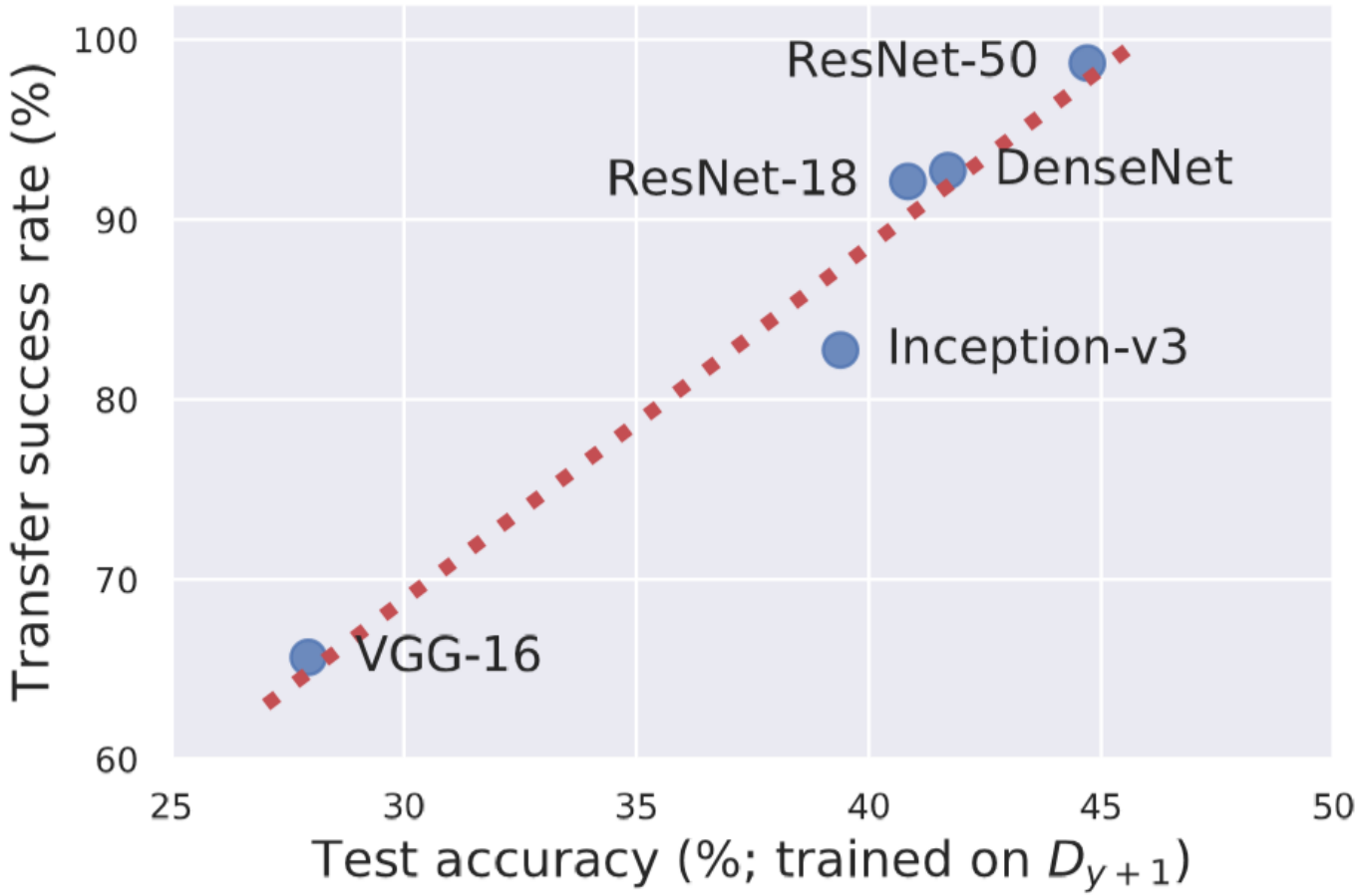


Figure: Examples from different datasets and performance of models trained on these datasets. Source [5]

Transferability

The tendency of different architecture to learn similar non-robust features correlates well with the transferability of adversarial examples between them.



Conclusion

1. Adversarial examples are the direct consequences of non-robust features present in the datasets.
2. Adversarial transferability occurs because different models learns the similar type of non-robust features from the dataset.
3. To get robust classifier we will need to enforce some prior on the learning method such that learning only relies on the robust features. This will also help with the interpretability of the deep neural networks.

Discussion I

There are some weakness of [5] that is worth discussing.

- Authors in [6] showed that adversarial examples can be computed without exploiting non-robust features. Hence the theory used by the authors in [5] is not completely sound.
- The authors have provided sound reasons connecting adversarial transferability to the existence of non-robust features. But there are cases, where the data distribution does not necessarily share non-robust features but still exhibit adversarial transferability [7].

Discussion II

- Finally, the authors have not accounted for common cases in the design and training of deep neural networks. For example, useful non-robust features can arise from the combinations of useful robust features and useless non-robust features. This shows that by just limiting their discussion to individual features in the feature set, the authors have left some important issues unaddressed.

Adversarial Learning and Wearable Computing

Problems and Future Works

In the following slides I will discuss some open problems that lies at the intersection of wearable computing and adversarial learning. I believe these problems are important to the discussion and are also related to the suggested work [5].

Development of Adversarial Attack Framework for Wearable Systems

- Wearable systems are closed systems in the sense that input-output pipeline is not exposed to foreign operators. The available adversarial attack methods will not work in this setting.
- Any adversarial attack to be successful in real-life wearable setting will need to take into account the intrinsic properties of these systems.

Time-Series and Human Comprehensibility

- The central premise of [5] depends on the human comprehensibility of robust features.
- But humans cannot understand time-series signals and in turn the idea of robust and non-robust features need better clarifications for time-series systems.

For example consider the images shown below:

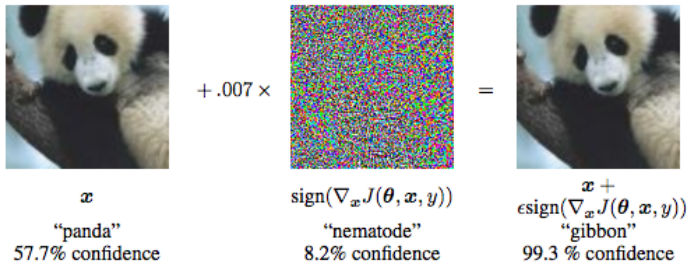


Figure: Image Adversarial Example. Source [1].

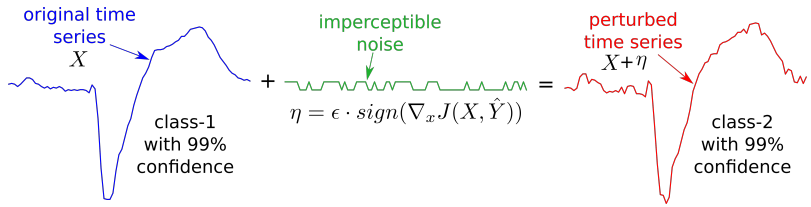


Figure: Time-Series Adversarial Example. Source [2].

Adversariality in Wearable Systems

- Theory that explains the reasons behind the existence of adversarial examples and transferability in time-series systems.
- Time-series systems does not share the idea of human comprehensibility that is present in image classification systems. Therefore the idea of robust and non-robust features presented in [5] will need modifications to conform with the characteristics of time-series systems.

Defense Methods

- We need to develop defense methods that takes into consideration the results presented in [5]. Specifically, the idea that adversarial examples are linked to the properties of the dataset and not the models.
- This is more applicable to time-series systems because of the temporal and spatial relationships in the data.

Thank You!

Questions ?

Non-robust Features

- Given a learned model $\theta = (\mu, \Sigma)$, θ defines an inner product over the input space represents how a change in input affects the features learned by the classifier.
- The authors uses the notion of metric to demonstrates how the misalignment in metrics learned by the model θ and used by the adversary l_2 results in non-robust features.
- Therefore, small change in the adversary's metric will cause large changes under data dependent notion of distance established by θ.

Adversarial Robustness

- With adversarial training the metric induced by the learned features mixes adversarial l_2 metric used to compute adversarial training and the metric induced by the features from the original dataset giving robust classifier.
- This means training with an l_2-bounded adversary prevents the classifier from relying heavily on features which induce a metric dissimilar to the l_2 metric.

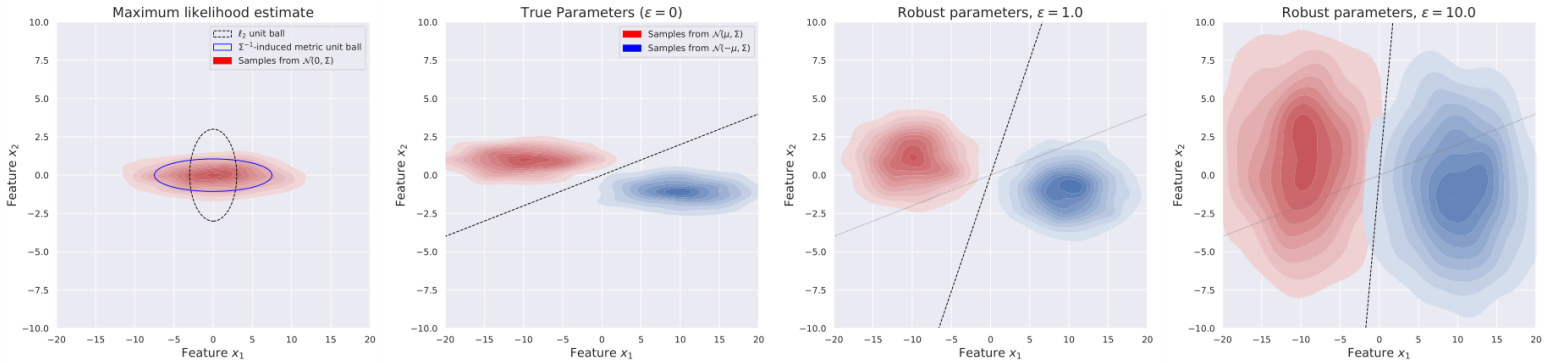


Figure 4: An empirical demonstration of the effect illustrated by Theorem 2—as the adversarial perturbation budget ε is increased, the learned mean $\boldsymbol{\mu}$ remains constant, but the learned covariance "blends" with the identity matrix, effectively adding more and more uncertainty onto the non-robust feature.

Figure: Source [5]

Generating Robust Dataset

GETROBUSTDATASET(D)

1. $C_R \leftarrow$ ADVERSARIALTRAINING(D)
   $g_R \leftarrow$ mapping learned by C_R from the input to the representation layer
2. $D_R \leftarrow \{\}$
3. For $(x, y) \in D$
   $x' \sim D$
   $x_R \leftarrow \arg\min_{z \in [0,1]^d} \|g_R(z) - g_R(x)\|_2$ # Solved using ℓ_2-PGD starting from x'
   $D_R \leftarrow D_R \bigcup \{(x_R, y)\}$
4. Return D_R

Figure 5: Algorithm to construct a "robust" dataset, by restricting to features used by a robust model.

Figure: Source [5]

Generating Non-Robust Dataset

GetNonRobustDataset(D, ε)

1. $D_{NR} \leftarrow \{\}$
2. $C \leftarrow$ StandardTraining(D)
3. For $(x, y) \in D$

   $t \overset{\text{uar}}{\sim} [C]$ # or $t \leftarrow (y+1) \mod C$

   $x_{NR} \leftarrow \min_{||x'-x|| \leq \varepsilon} L_C(x', t)$ # Solved using ℓ_2 PGD

   $D_{NR} \leftarrow D_{NR} \bigcup \{(x_{NR}, t)\}$
4. Return D_{NR}

Figure 6: Algorithm to construct a dataset where input-label association is based entirely on non-robust features.

Figure: Source [5]

References

References I


I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *CoRR*, vol. abs/1412.6572, 2014.


H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Adversarial attacks on deep neural networks for time series classification," *2019 International Joint Conference on Neural Networks (IJCNN)*, Jul 2019.


K. Rajaratnam and J. K. Kalita, "Noise flooding for detecting audio adversarial examples against automatic speech recognition," *2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pp. 197–201, 2018.


I. Goodfellow, P. McDaniel, and N. Papernot, "Making machine learning robust against adversarial inputs," *Commun. ACM*, vol. 61, p. 56–66, June 2018.


A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," in *Advances in Neural Information Processing Systems 32*, pp. 125–136, Curran Associates, Inc., 2019.


P. Nakkiran, "A discussion of 'adversarial examples are not bugs, they are features': Adversarial examples are just bugs, too," *Distill*, 2019.
https://distill.pub/2019/advex-bugs-discussion/response-5.


M. Naseer, S. H. Khan, H. Khan, F. S. Khan, and F. Porikli, "Cross-domain transferability of adversarial perturbations," 2019.