Robustness in AI

A brief history and introduction

Aditya Prasad

January 30, 2023

Context and Motivation

- 2012 – Rise of neural networks (NNs) for computer vision (ImageNet, AlexNet)
  - Research in NNs for computer vision was influenced by prior (non-NN) computer vision research

Context and Motivation

- 2012 – Rise of neural networks (NNs) for computer vision (ImageNet, AlexNet)
  - Research in NNs for computer vision was influenced by prior (non-NN) computer vision research
- One obvious intuition – anything that generalizes well should (at a minimum!) do well on points "close" to the training set (i.e., *local generalization*)
  - We'll see that this intuition fails miserably!

Context and Motivation

- 2012 – Rise of neural networks (NNs) for computer vision (ImageNet, AlexNet)
  - Research in NNs for computer vision was influenced by prior (non-NN) computer vision research
- One obvious intuition – anything that generalizes well should (at a minimum!) do well on points "close" to the training set (i.e., *local generalization*)
  - We'll see that this intuition fails miserably!
- Lots of emphasis on understanding key subsets of neural networks, called features

Geometric Intuition

Table of Contents

Table of Contents

Notation

- We'll focus primarily on image classification using deep neural networks (DNNs).

Notation

- We'll focus primarily on image classification using deep neural networks (DNNs).
- $x \in \mathbb{R}^m$: input image.

Notation

- We'll focus primarily on image classification using deep neural networks (DNNs).
- $x \in \mathbb{R}^m$: input image.
- $\phi(x)$: activation values for some layer on input x.

Table of Contents

- Feature extraction on a neural network trained on MNIST.
- Fix an activation unit ϕ and consider

$$x' = \arg\max_{x \in \mathcal{I}} \langle \phi(x), e_i \rangle.$$

[0] This chapter is based on content from [3].

Part 1 - Understanding NNs

- Feature extraction on a neural network trained on MNIST.
- Fix an activation unit ϕ and consider

$$x' = \arg\max_{x \in \mathcal{I}} \langle \phi(x), e_i \rangle.$$

- For any activation unit, consider the images that are most in-line with basis vectors.
  - These images should have visually perceptive commonalities.

[0] This chapter is based on content from [3].

Part 1 - Understanding NNs

- Feature extraction on a neural network trained on MNIST.
- Fix an activation unit ϕ and consider

$$x' = \arg\max_{x \in \mathcal{I}} \langle \phi(x), e_i \rangle.$$

- For any activation unit, consider the images that are most in-line with basis vectors.
  - These images should have visually perceptive commonalities.
- Understand the structure or pattern that the activation unit ϕ is "extracting."

[0] This chapter is based on content from [3].

(a) Unit sensitive to lower round stroke.

(b) Unit sensitive to upper round stroke, or lower straight stroke.

(c) Unit senstive to left, upper round stroke.

(d) Unit senstive to diagonal straight stroke.

Figure 1: An MNIST experiment. The figure shows images that maximize the activation of various units (maximum stimulation in the natural basis direction). Images within each row share semantic properties.

[0] Figure from [3].

Part 1 - Understanding NNs

- However, performing the same analysis for a random vector v yields similarly interpretable results.

[0] Figure from [3].

Part 1 - Understanding NNs

- However, performing the same analysis for a random vector v yields similarly interpretable results.

(a) Direction sensitive to upper straight stroke, or lower round stroke.

(b) Direction sensitive to lower left loop.

(c) Direction senstive to round top stroke.

(d) Direction sensitive to right, upper round stroke.

Figure 2: An MNIST experiment. The figure shows images that maximize the activations in a random direction (maximum stimulation in a random basis). Images within each row share semantic properties.

[0] Figure from [3].

Part 1 - Understanding NNs

- However, performing the same analysis for a random vector v yields similarly interpretable results.

(a) Direction sensitive to upper straight stroke, or lower round stroke.

(b) Direction sensitive to lower left loop.

(c) Direction senstive to round top stroke.

(d) Direction sensitive to right, upper round stroke.

Figure 2: An MNIST experiment. The figure shows images that maximize the activations in a random direction (maximum stimulation in a random basis). Images within each row share semantic properties.

- The standard basis e_i is no better than a random basis for understanding $\phi(x)$!

[0]Figure from [3].

Part 2 - Adversarial Examples

- Recall the intuition of *local generalization* - models that generalize well globally also generalize well locally.
- Implied a sense of "smoothness" (similar to Lipschitz continuity) of the model that held for many models prior to NNs.

Part 2 - Adversarial Examples

- Recall the intuition of *local generalization* - models that generalize well globally also generalize well locally.
- Implied a sense of "smoothness" (similar to Lipschitz continuity) of the model that held for many models prior to NNs.
- This intuition does not hold for NNs!
- There exist "adversarial" examples – inputs which are ϵ-close to a training point and misclassified by the model.

Part 2 - Adversarial Examples

Let $f : \mathbb{R}^m \to \{1, ..., k\}$ be a classifier mapping pictures to labels. Want to solve the following optimization problem $D(x, l)$:

$$\begin{aligned} \text{Minimize: } & ||r||_2 \\ \text{Subject to: } & f(x + r) = l \\ & x + r \in [0, 1]^m \end{aligned}$$

Part 2 - Adversarial Examples

Let $f : \mathbb{R}^m \to \{1, ..., k\}$ be a classifier mapping pictures to labels. Want to solve the following optimization problem $D(x, l)$:

$$\begin{aligned} \text{Minimize: } & ||r||_2 \\ \text{Subject to: } & f(x + r) = l \\ & x + r \in [0, 1]^m \end{aligned}$$

- Finds smallest perturbation vector r that causes f to classify input x as label l.
- Importantly, we get to choose a different r for each x.

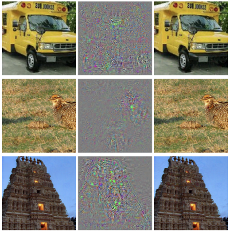
Part 2 - Adversarial Examples

Let $f : \mathbb{R}^m \to \{1, ..., k\}$ be a classifier mapping pictures to labels. Want to solve the following optimization problem $D(x, l)$:

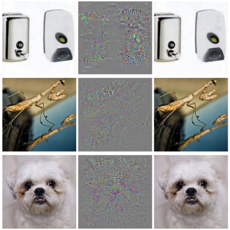
$$\begin{aligned} \text{Minimize: } & ||r||_2 \\ \text{Subject to: } & f(x+r) = l \\ & x + r \in [0,1]^m \end{aligned}$$

- Finds smallest perturbation vector r that causes f to classify input x as label l.
- Importantly, we get to choose a different r for each x.
- This is hard problem – we approximate.
- For $c > 0$, solve: $\min_r [c|r| + \text{loss}_f(x + r, l)]$ subject to $x + r \in [0,1]^m$.

Part 2 - Adversarial Examples



(a)



(b)

Figure 5: Adversarial examples generated for AlexNet [9].(Left) is a correctly predicted sample, (center) difference between correct image, and image predicted incorrectly magnified by 10x (values shifted by 128 and clamped), (right) adversarial example. All images in the right column are predicted to be an *"ostrich, Struthio camelus"*. Average distortion based on 64 examples is 0.006508. Plase refer to `http://goo.gl/huaGPb` for full resolution images. The examples are strictly randomly chosen. There is not any postselection involved.

[0] Figure from [3].

Part 2 - Adversarial Examples

Rest of paper:

- Showed that adversarial inputs are universal – they are not the result of any particular model overfitting.
- Provided a bound on the Lipschitz constant on each layer of the NN.

Part 2 - Adversarial Examples

Rest of paper:

- Showed that adversarial inputs are universal – they are not the result of any particular model overfitting.
- Provided a bound on the Lipschitz constant on each layer of the NN.
- In DNNs (ImageNet), instabilities can appear as early as the first layer.

Key Takeaways

Key takeaways:

Key Takeaways

Key takeaways:

- Feature extraction is an uninformative method of analysis.

Key Takeaways

Key takeaways:

- Feature extraction is an uninformative method of analysis.
- For any NNs, there exist adversarial examples that are close to training examples which are confidently misclassified.

Key Takeaways

Key takeaways:

- Feature extraction is an uninformative method of analysis.
- For any NNs, there exist adversarial examples that are close to training examples which are confidently misclassified.
- Implies that NNs are not Lipschitz continuous for small constants.

Table of Contents

Universal perturbation

- Let μ be a distribution of images in $\mathbb{R}^d$ and $\hat{k}$ be a classifier that assigns $x \in \mathbb{R}^d$ a label $\hat{k}(x)$.

[0]This chapter is based on content from [2].

Universal perturbation

- Let μ be a distribution of images in $\mathbb{R}^d$ and $\hat{k}$ be a classifier that assigns $x \in \mathbb{R}^d$ a label $\hat{k}(x)$.
- Universal perturbation - $v \in \mathbb{R}^d$ such that

$$\hat{k}(x + v) \neq \hat{k}(x) \text{ for most } x \sim \mu.$$

- Note that v is universal and image agnostic.

[0] This chapter is based on content from [2].

Universal perturbation

- Let μ be a distribution of images in $\mathbb{R}^d$ and $\hat{k}$ be a classifier that assigns $x \in \mathbb{R}^d$ a label $\hat{k}(x)$.
- Universal perturbation - $v \in \mathbb{R}^d$ such that

$$\hat{k}(x+v) \neq \hat{k}(x) \text{ for most } x \sim \mu.$$

- Note that v is universal and image agnostic.
- Want v to be imperceptible according to a p-norm of user's choice and misclassify most examples:

$$||v||_p \leq \xi$$
$$\mathbb{P}_{x\sim\mu}(\hat{k}(x+v) \neq \hat{k}(x)) \geq 1-\delta$$

[0]This chapter is based on content from [2].

Universal Perturbation - Geometrically

Algorithm

- To find universal perturbation v, set $v = \vec{0}$ and find an image x_i that is correctly classified, then set

Algorithm

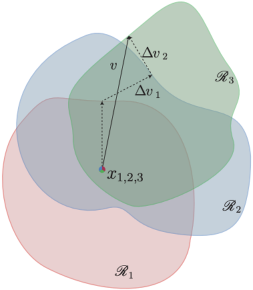
- To find universal perturbation v, set $v = \vec{0}$ and find an image x_i that is correctly classified, then set

$$\Delta v_i \leftarrow \arg\min_r ||r||_2 \text{ s.t. } \hat{k}(x_i + v + r) \neq \hat{k}(x_i), \text{ (direction)}$$

- Set $v \leftarrow v + \Delta v_i$ normalized so p-norm is small.

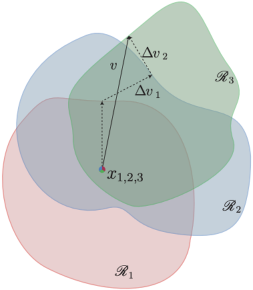
- Repeat until empirical "fooling rate" is larger than $1 - \delta$.

Algorithm Intuition



[0]Figure from [2].

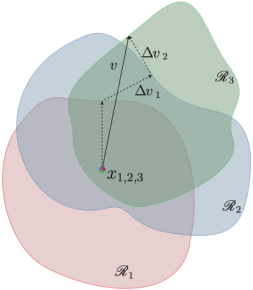
Algorithm Intuition



- Find a correctly classified example and add smallest perturbation that misclassifies it.

[0] Figure from [2].

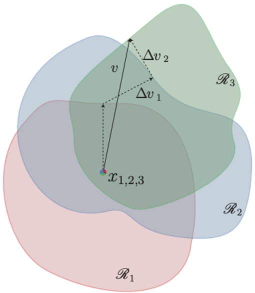
Algorithm Intuition



- Find a correctly classified example and add smallest perturbation that misclassifies it.
- Add that perturbation to current perturbation.

[0] Figure from [2].

Algorithm Intuition



- Find a correctly classified example and add smallest perturbation that misclassifies it.
- Add that perturbation to current perturbation.
- Rinse and repeat until model is fooled whp.

[0] Figure from [2].

Properties of Universal Adversarial Perturbations

- Showed that UAPs can be found across many sets of data points and for very different architectures.

Properties of Universal Adversarial Perturbations

- Showed that UAPs can be found across many sets of data points and for very different architectures.
- Showed empirically that it requires relatively few examples to obtain UAP vector.

Properties of Universal Adversarial Perturbations

- Showed that UAPs can be found across many sets of data points and for very different architectures.
- Showed empirically that it requires relatively few examples to obtain UAP vector.
- Showed that UAP fooled models on approx 80% of train and validation data.

Properties of Universal Adversarial Perturbations

- Showed that UAPs can be found across many sets of data points and for very different architectures.
- Showed empirically that it requires relatively few examples to obtain UAP vector.
- Showed that UAP fooled models on approx 80% of train and validation data.
- Interestingly, if the model is retrained to learn on images with UAP_1, it still will do poorly on new UAP_2 (still fooled on 75% of data).

Properties of Universal Adversarial Perturbations

- Showed that UAPs can be found across many sets of data points and for very different architectures.
- Showed empirically that it requires relatively few examples to obtain UAP vector.
- Showed that UAP fooled models on approx 80% of train and validation data.
- Interestingly, if the model is retrained to learn on images with UAP_1, it still will do poorly on new UAP_2 (still fooled on 75% of data).
    - This process can be repeated an arbitrary amount of times and will not improve accuracy of fooled model.

Misclassification graph

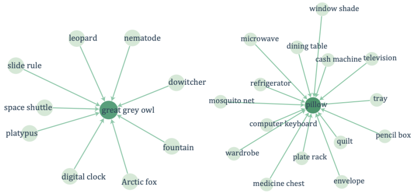


Figure 7: Two connected components of the graph $G = (V, E)$, where the vertices are the set of labels, and directed edges $i \to j$ indicate that most images of class i are fooled into class j.

Figure: Misclassification graph — (u, v) means that most examples of u are fooled to v

[0]Figure from [2].

Misclassification graph

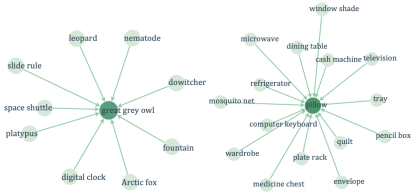


Figure 7: Two connected components of the graph $G = (V, E)$, where the vertices are the set of labels, and directed edges $i \to j$ indicate that most images of class i are fooled into class j.

Figure: Misclassification graph — (u, v) means that most examples of u are fooled to v

- When models are fooled, they have a disproportionately high chance to pick so-called *dominant labels*.

[0]Figure from [2].

Misclassification graph

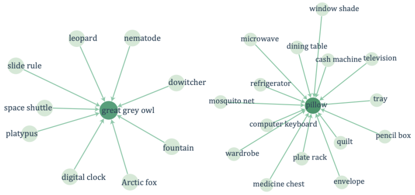


Figure 7: Two connected components of the graph $G = (V, E)$, where the vertices are the set of labels, and directed edges $i \to j$ indicate that most images of class i are fooled into class j.

Figure: Misclassification graph — (u, v) means that most examples of u are fooled to v

- When models are fooled, they have a disproportionately high chance to pick so-called *dominant labels*.
- Conjectured that dominant labels occupy large regions of image space.

[0]Figure from [2].

Table of Contents

Need for Robustness

[0] This chapter is based on content from [1].

Need for Robustness

- Clearly, DNNs are not locally generalizing.

[0]This chapter is based on content from [1].

Need for Robustness

- Clearly, DNNs are not locally generalizing.
- Are small perturbations useful? We limit ourselves to “imperceptible" changes. Why?

[0] This chapter is based on content from [1].

Need for Robustness

- Clearly, DNNs are not locally generalizing.
- Are small perturbations useful? We limit ourselves to “imperceptible" changes. Why?
- In the real world, we may not need changes to be imperceptible in order to preserve meaning.

[0] This chapter is based on content from [1].

Need for Robustness

- Clearly, DNNs are not locally generalizing.
- Are small perturbations useful? We limit ourselves to “imperceptible" changes. Why?
- In the real world, we may not need changes to be imperceptible in order to preserve meaning.
- Robustness* - if an input is modified in a way that preserves semantics, classification should not change.

[0] This chapter is based on content from [1].

Need for Robustness

- Clearly, DNNs are not locally generalizing.
- Are small perturbations useful? We limit ourselves to "imperceptible" changes. Why?
- In the real world, we may not need changes to be imperceptible in order to preserve meaning.
- Robustness* - if an input is modified in a way that preserves semantics, classification should not change.
- We have seen that classical analysis techniques will not work. How to proceed?

[0] This chapter is based on content from [1].

Robustness as a Game

- The main goal of this position paper is to phrase robustness as a game with two parties, the attacker (adversary) and the classifier.
    - Many possible rules to this game!

Robustness as a Game

- The main goal of this position paper is to phrase robustness as a game with two parties, the attacker (adversary) and the classifier.
  - Many possible rules to this game!
- Attacker's goal:

Robustness as a Game

- The main goal of this position paper is to phrase robustness as a game with two parties, the attacker (adversary) and the classifier.
  - Many possible rules to this game!
- Attacker's goal:
  - *Targeted attack* – Induce a specific error.
  - *Untargeted attack* – Induce any error.

Robustness as a Game

- The main goal of this position paper is to phrase robustness as a game with two parties, the attacker (adversary) and the classifier.
    - Many possible rules to this game!
- Attacker's goal:
    - *Targeted attack* – Induce a specific error.
    - *Untargeted attack* – Induce any error.
- Attacker's knowledge:

Robustness as a Game

- The main goal of this position paper is to phrase robustness as a game with two parties, the attacker (adversary) and the classifier.
    - Many possible rules to this game!
- Attacker's goal:
    - *Targeted attack* – Induce a specific error.
    - *Untargeted attack* – Induce any error.
- Attacker's knowledge:
    - *Whitebox setting* – Attacker has full knowledge of classifer internals and training data.
    - *Blackbox setting* – Attacker only can ask classifier to label inputs.

Robustness as a Game

- The main goal of this position paper is to phrase robustness as a game with two parties, the attacker (adversary) and the classifier.
  - Many possible rules to this game!
- Attacker's goal:
  - *Targeted attack* – Induce a specific error.
  - *Untargeted attack* – Induce any error.
- Attacker's knowledge:
  - *Whitebox setting* – Attacker has full knowledge of classifer internals and training data.
  - *Blackbox setting* – Attacker only can ask classifier to label inputs.
- Who goes first? Is the game repeated?

Power of Adversary

- Does the attacker get to choose the input?
- In what ways can an attacker actually attack?

Power of Adversary

- Does the attacker get to choose the input?
- In what ways can an attacker actually attack?
  - *Indistinguishable perturbation* – No control over input but can perturb it a little bit.

Power of Adversary

- Does the attacker get to choose the input?
- In what ways can an attacker actually attack?
  - *Indistinguishable perturbation* – No control over input but can perturb it a little bit.
  - *Content-preserving perturbation* – No limit on perturbation size as long as content is preserved.

Power of Adversary

- Does the attacker get to choose the input?
- In what ways can an attacker actually attack?
  - *Indistinguishable perturbation* – No control over input but can perturb it a little bit.
  - *Content-preserving perturbation* – No limit on perturbation size as long as content is preserved.
  - *Non-suspicious input* – attacker can design any innocent-looking input.

Power of Adversary

- Does the attacker get to choose the input?
- In what ways can an attacker actually attack?
  - *Indistinguishable perturbation* – No control over input but can perturb it a little bit.
  - *Content-preserving perturbation* – No limit on perturbation size as long as content is preserved.
  - *Non-suspicious input* – attacker can design any innocent-looking input.
  - *Content-constrained input* – attacker can design an input as long as it contains a specific content payload.

Power of Adversary

- Does the attacker get to choose the input?
- In what ways can an attacker actually attack?
  - *Indistinguishable perturbation* – No control over input but can perturb it a little bit.
  - *Content-preserving perturbation* – No limit on perturbation size as long as content is preserved.
  - *Non-suspicious input* – attacker can design any innocent-looking input.
  - *Content-constrained input* – attacker can design an input as long as it contains a specific content payload.
  - *Unconstrained input*

Example

Figure 1: An example of image spam shown in Biggio and Roli [77]. Note the notion of a "starting point" does not apply here, instead the entire image is crafted from scratch by the attacker to avoid statistical detection. It is not of the form of applying a small or imperceptible perturbation to random image from the training distribution. Thanks to Battista Biggio for permission to use this image.

[0] Figure from [1].

Example

Figure 1: An example of image spam shown in Biggio and Roli [77]. Note the notion of a "starting point" does not apply here, instead the entire image is crafted from scratch by the attacker to avoid statistical detection. It is not of the form of applying a small or imperceptible perturbation to random image from the training distribution. Thanks to Battista Biggio for permission to use this image.

- Model on social media that blocks investing advice.

[0] Figure from [1].

Example

Figure 1: An example of image spam shown in Biggio and Roli [77]. Note the notion of a "starting point" does not apply here, instead the entire image is crafted from scratch by the attacker to avoid statistical detection. It is not of the form of applying a small or imperceptible perturbation to random image from the training distribution. Thanks to Battista Biggio for permission to use this image.

- Model on social media that blocks investing advice.
- No notion of starting point and clearly want to be robust to perceptible changes.

[0] Figure from [1].

Other examples

Other examples

- The Stop Sign Attack

Other examples

- The Stop Sign Attack
- Evading Malware Detection

Other examples

- The Stop Sign Attack
- Evading Malware Detection
- Fooling Facial Recognition

Other examples

- The Stop Sign Attack
- Evading Malware Detection
- Fooling Facial Recognition
- Check Fraud

Other examples

- The Stop Sign Attack
- Evading Malware Detection
- Fooling Facial Recognition
- Check Fraud
- etc.

Rest of paper

- Covers other forms of attacks and attack scenarios.
- Gives more real world examples.
- Justifies robustness research.

Caution: Small norm $\neq$ imperceptible

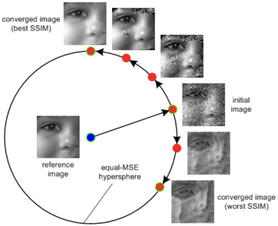


Figure 2: Images equally far away from a reference image in the l_2 sense can be dramatically different in perceived distance. Figure due to Wang and Bovik [97].

- Warning: ℓ_p norms may not align with our intuition!

[0]Figure from [1].

Table of Contents

Bibliography I

Justin Gilmer et al. "Motivating the Rules of the Game for Adversarial Example Research". In: *CoRR* abs/1807.06732 (2018). arXiv: 1807.06732. URL: http://arxiv.org/abs/1807.06732.

Seyed-Mohsen Moosavi-Dezfooli et al. "Universal Adversarial Perturbations". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 86–94. DOI: 10.1109/CVPR.2017.17. URL: https://doi.org/10.1109/CVPR.2017.17.

Christian Szegedy et al. "Intriguing properties of neural networks". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2014. URL: http://arxiv.org/abs/1312.6199.

Thank you!

[0] Special thanks to Julian Asilis for helping to create this presentation!