AI Security

Adversarial Examples

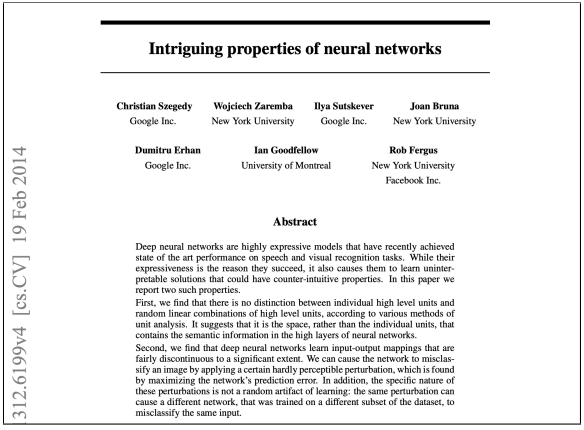
Sangdon Park

POSTECH

Outline

1. Introduction
2. White-box Attack Algorithms
3. Transfer Attack Algorithms
4. Black-box Attack Algorithms
5. Interpretation
6. Conclusion

Discovery of Adversarial Examples



Intriguing properties of neural networks

Christian Szegedy Google Inc.

Wojciech Zaremba New York University

Ilya Sutskever Google Inc.

Joan Bruna New York University

Dumitru Erhan Google Inc.

Ian Goodfellow University of Montreal

Rob Fergus New York University
Facebook Inc.

Abstract

Deep neural networks are highly expressive models that have recently achieved state of the art performance on speech and visual recognition tasks. While their expressiveness is the reason they succeed, it also causes them to learn uninterpretable solutions that could have counter-intuitive properties. In this paper we report two such properties.

First, we find that there is no distinction between individual high level units and random linear combinations of high level units, according to various methods of unit analysis. It suggests that it is the space, rather than the individual units, that contains the semantic information in the high layers of neural networks.

Second, we find that deep neural networks learn input-output mappings that are fairly discontinuous to a significant extent. We can cause the network to misclassify an image by applying a certain hardly perceptible perturbation, which is found by maximizing the network's prediction error. In addition, the specific nature of these perturbations is not a random artifact of learning: the same perturbation can cause a different network, that was trained on a different subset of the dataset, to misclassify the same input.

- One of 35 papers, presented at the 2nd International Conference on Learning Representations (ICLR) in 2014.

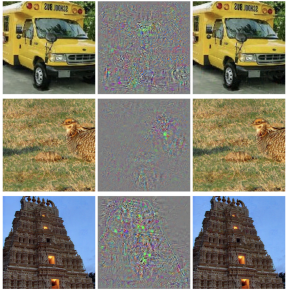
Adversarial Examples

Definition

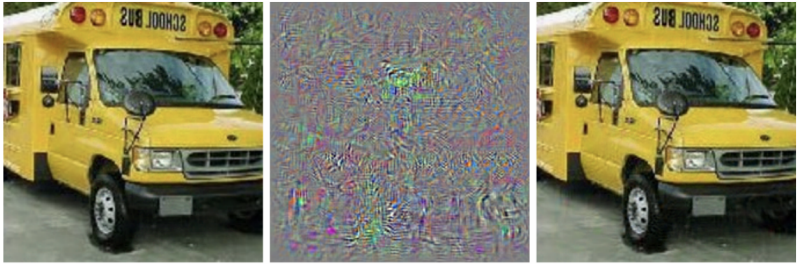
An *adversarial example* x_{adv} of an example x for a classifier $f : \mathcal{X} \to \mathcal{Y}$ is an "imperceptibly" perturbed example where a predicted label changes before and after adding the perturbation, *i.e.*,

$$x_{\text{adv}} \coloneqq x + \delta \quad \text{subject to} \quad \|\delta\| \leq \varepsilon \text{ and } f(x) \neq f(x_{\text{adv}}),$$

where $\mathcal{X}$ is a set of examples and $\mathcal{Y}$ is a set of labels.

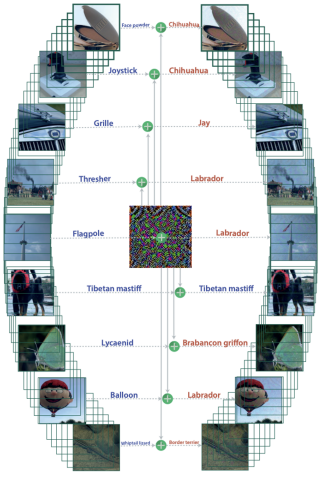


Why Intriguing?



1. The adversarial perturbation is "imperceptible".
   - ε is often very small, *e.g.*, $\varepsilon = \frac{8}{255} \approx 0.03$.
   - Note that adversarial examples with a larger perturbation provides trivial results.
2. The adversarial examples are mostly transferable.
   - If x_{adv} is an adversarial example for f, *i.e.*, $f(x_{\text{adv}}) \neq f(x)$, then $f'(x_{\text{adv}}) \neq f'(x)$.
   - No theoretical justification but empirically verifiable.

Why Intriguing?



3. universal adversarial perturbation (*i.e.*, one adversarial perturbation that makes the most images being mis-classified) exists [Moosavi-Dezfooli et al., 2017].

Goal of This Class

Our Key Question

How to find an adversarial example?

The equivalent and related questions include the following:

- How to attack classifiers?
- How to find the vulnerabilities of classifiers?
- How to find an adversarial examples without having access to a classifier?
- What's the reason for the existence of adversarial examples?

Outline

1. Introduction
2. White-box Attack Algorithms
3. Transfer Attack Algorithms
4. Black-box Attack Algorithms
5. Interpretation
6. Conclusion

How to find Adversarial Examples?

All you need is to solve the following optimization problem.

$$\delta^* = \max_{\|\delta\| \leq \varepsilon} \ell(f, x + \delta, y)$$

- $x \in \mathcal{X}$: an example
- $y \in \mathcal{Y}$: a label
- $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$: a (soft) classifier
- $\varepsilon \geq 0$: a maximum perturbation
- $\|\cdot\|$: a norm (but often ∞- or 2-norm)
- ℓ: a loss function (*e.g.*, cross-entropy loss)
- Why is solving this maximization hard?

L-BFSG Method

This is the first attack method.

L-BFSG [Szegedy et al., 2014]

$$\min_{\delta} \; c\|\delta\|_2 + \ell(f, x+\delta, y_{\text{target}}) \quad \text{subj. to} \quad x+\delta \in [0,1]^m$$

- c: a hyper-parameter
- Broyden–Fletcher–Goldfarb–Shanno(BFGS) method: an iterative method for solving unconstrained non-linear optimization problems (= an off-the-shelf optimizer).
- L-BFGS: a limited-memory version of BFGS (= an off-the-shelf optimizer)
- Find a perturbation δ that is small but changes the prediction to the target label y_{target}, while the perturbed examples stays in a valid domain (*e.g.*, an image domain).
  - this is also called a targeted attack.
  - How can we find a small perturbation?
  - What if y_{target} is the same as the prediction without perturbation, *i.e.*, $y_{\text{target}} = \arg\max_y f(x, y)$?

FGSM: Fast Gradient Sign Method

This is the most famous and simple attack method.

FGSM [Goodfellow et al., 2015]

$$\max_{\|\delta\|_\infty \leq \varepsilon} \ell(f, x + \delta, y)$$

- This explicitly controls the perturbation size by ε.
- y: y is often the true label for x.
- This is an un-targeted attack.
- How to solve this?

FGSM: A Closed-form Solution

FGSM

$$\varepsilon \cdot \text{sign}(\nabla_x \ell(f, x, y)) = \arg \max_{\|\delta\|_\infty \leq \varepsilon} \ell(f, x + \delta, y)$$

- Intuition:
  - the partial derivative of loss w.r.t. an example = the change of loss w.r.t. the perturbation
  - only consider the sign of the change.
  - amplify the magnitude by ε.
- This makes some sense but is this an optimal solution under some conditions?

Taylor Series I

The Taylor series represents a function f as an infinite sum of terms.

Definition (Taylor Series)

For a smooth, *i.e.,* infinitely differentiable, function $f : \mathbb{R} \to \mathbb{R}$, the *Taylor series* of f at x_0 is defined as

$$f(x) = \lim_{K \to \infty} T_K(x) := \sum_{k=0}^{K} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k = f(x_0) + \frac{f'(x_0)}{1!}(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \ldots,$$

where $f^{(k)}(x_0)$ is the kth derivative of f at x_0.

Taylor Series II

- An example: the exponential function e^x at $x_0 = 0$ is represented via the Taylor series, *i.e.*,

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}.$$

See animation:
`https://en.wikipedia.org/wiki/Taylor_series#/media/File:Exp_series.gif`.

Taylor Series III

- How to use it for approximation? Use the summation over a first few terms to approximate the original function f.
  - An example: The first order Taylor expansion $T_1(x)$, *i.e.*,

$$f(x) \approx T_1(x) = f(x_0) + f'(x_0)(x - x_0)$$

    - ⋆ Useful to approximate around x_0
    - ⋆ Useful when second and higher order derivatives are computationally expansive.

FGSM: Optimality Analysis [Lyu et al., 2015] I

Approximate the loss via the first order Talyor expansion.

the first order approximation

$$\tilde{\ell}(x_0 + \delta) \approx \tilde{\ell}(x_0) + \nabla\tilde{\ell}(x_0)\delta^T$$

- Let $\tilde{\ell}(x) \coloneqq \ell(f, x, y)$.
- The original first order Taylor expansion of $\tilde{\ell}$ at x_0:

$$\tilde{\ell}(x) \approx \tilde{\ell}(x_0) + \nabla\tilde{\ell}(x_0)(x - x_0)^T$$

- Letting $x = x_0 + \delta$ as we approximate loss around an example x_0, we have

$$\tilde{\ell}(x_0 + \delta) \approx \tilde{\ell}(x_0) + \nabla\tilde{\ell}(x_0)^T\delta.$$

FGSM: Optimality Analysis [Lyu et al., 2015] II

- Our goal:

$$\max_{\|\delta\|_\infty \leq \varepsilon} \tilde{\ell}(x_0 + \delta) \approx \max_{\|\delta\|_\infty \leq \varepsilon} \tilde{\ell}(x_0) + \nabla \tilde{\ell}(x_0)^T \delta$$

  - Why do we need this approximation?

- How to solve the approximated maximization?
  - Is it easy to solve the maximization?
  - Use any optimization package?

FGSM: Optimality Analysis [Lyu et al., 2015] III

We have a closed form solution!

A closed-form optimal solution

$$\varepsilon \cdot \text{sign}(\nabla \tilde{\ell}(x_0)) = \arg \max_{\|\delta\|_\infty \leq \varepsilon} \tilde{\ell}(x_0) + \nabla \tilde{\ell}(x_0)^T \delta$$

- We need to solve the following where $g := \nabla \tilde{\ell}(x_0)$:

$$\max_{\|\delta\|_\infty \leq \varepsilon} g^T \delta$$

- To maximize the inner product, we need to maximize each element of δ toward the direction of the corresponding element of g, *i.e.*, $\delta \propto \text{sign}(g)$.
- δ has a constraint via the ℓ_∞ norm, where the maximum value of an element of δ is ε; thus, we have our final solution, *i.e.*, $\delta = \varepsilon \cdot \text{sign}(g)$

FGSM: Generalization I

Gradient Regularization Family for Adversarial Examples [Lyu et al., 2015]

$$\delta_p = \varepsilon \cdot \text{sign}(\nabla_x \ell(f, x, y)) \left(\frac{\nabla_x \ell(f, x, y)}{\|\nabla_x \ell(f, x, y)\|_{p^*}} \right)^{\frac{1}{p-1}}$$

- p^* us the dual of p, *i.e.*, $\frac{1}{p^*} + \frac{1}{p} = 1$.
- We assume a general adversarial perturbation with a ℓ_p norm, *i.e.*,

$$\max_{\|\delta\|_p \leq \varepsilon} \ell(f, x + \delta, y)$$

- It is proven that δ_p is optimal for adversarial perturbation with a ℓ_p norm.

FGSM: Generalization II

- Check for FGSM, *i.e.*, $p = \infty$:

$$\begin{aligned}
\lim_{p\to\infty} \delta_p &= \varepsilon \cdot \operatorname{sign}(\nabla_x \ell(f, x, y)) \left(\frac{\nabla_x \ell(f, x, y)}{\|\nabla_x \ell(f, x, y)\|_{p^*}} \right)^{\frac{1}{p-1}} \\
&= \varepsilon \cdot \operatorname{sign}(\nabla_x \ell(f, x, y)) \left(\frac{\nabla_x \ell(f, x, y)}{\|\nabla_x \ell(f, x, y)\|_{p^*}} \right)^{0} \\
&= \underbrace{\varepsilon \cdot \operatorname{sign}(\nabla_x \ell(f, x, y))}_{\text{solution for FSGM!}}
\end{aligned}$$

FGSM Summary I

- Assumption on adversaries (= threat models, attacker's capabilities):
  1. an attacker has access to target model parameters, *e.g.,* the duplication of f is given to the attacker.
     - ⋆ This is why the FGSM is called a white-box attack.
  2. an attacker cannot add large perturbations, *i.e.,* $\|\delta\| \leq \varepsilon$.
     - ⋆ If the attacker add large perturbations, nearly no way to defend against it.
  3. an attacker has a capability to modify the input of a target model.
     - ⋆ This is why physical adversarial examples or universal adversaral examples are required.
  4. an attacker knows the true label of a target example x.
     - ⋆ It is okay to use a model's prediction instead of the true label.

FGSM Summary II

- Attacker's goal: change the prediction to something else, *i.e.*,

$$\max_{\|\delta\|_\infty \leq \varepsilon} \ell(f, x + \delta, y)$$

- The FSGM method finds the adversarial perturbation via $\delta = \varepsilon \cdot \text{sign}(\nabla_x \ell(f, x, y))$.
  - This is a closed-form solution so it is fast to generate an adversarial perturbation.
  - This is possible under the following assumption:

FSGM Assumption

The FGSM assumes that the loss landscape is linear w.r.t. the input.

- Is this assumption okay?

"Iterative" FGSM = PGD

Projected Gradient Descent (PGD) Method [Madry et al., 2017]

$$x^{(t+1)} = \Pi_{x+\mathcal{S}}\left(x^{(t)} + \alpha \cdot \mathrm{sign}(\nabla_{x^{(t)}} \ell(f, x^{(t)}, y)\right),$$

where $x^{(0)} \coloneqq x$.

- $x^{(t)}$: the t-th adversarial example.
- $\mathcal{S}$: a set of perturbations, *e.g.*, $\mathcal{S}_{\varepsilon,\infty} \coloneqq \{\delta \mid \|\delta\|_\infty \leq \varepsilon\}$
- $\Pi_{x+\mathcal{S}}(z)$: the projection of z to the closest point in the set $x + \mathcal{S}$
  - How to implement the projection w.r.t. $\mathcal{S}_{\varepsilon,\infty}$?
- α: step size
- "Iterative" FSGM is originally proposed by Google [Kurakin et al., 2017] prior to PGD.
- The PGD is multi-step attack, while FGSM is one-step attack.

Sparse Adversarial Examples

SparseFool [Modas et al., 2019]

$$\min_{\delta} \|\delta\|_1 \quad \text{subj. to} \quad \hat{y}(x+\delta) \neq \hat{y}(x)$$

- The proposed method heuristically solves the minimization problem.
- Why does ℓ_1 norm help to find a sparse perturbation?
- Check out One-pixel Attack

Universal Adversarial Perturbation

universal adversarial perturbation [Moosavi-Dezfooli et al., 2017]

$$\text{find } \delta \quad \text{subj. to} \quad \|\delta\|_p \leq \varepsilon \text{ and } \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\left(\hat{y}(x_i + \delta) \neq \hat{y}(x_i)\right) \geq 1 - \delta$$

- The proposed algorithm heuristically finds δ, *i.e.,*
  1. for each x_i
  2. if $\hat{y}(x_i + \delta) = \hat{y}(x_i)$, find a minimal perturbation to change the decision, *i.e.,*

$$\Delta\delta_i \leftarrow \arg\min_{\delta_i} \|\delta_i\|_2 \text{ subj. to } \hat{y}(x_i + \delta + \delta_i) \neq \hat{y}(x_i)$$

  3. Update the perturbation, *i.e.,*

$$\delta \leftarrow \Pi_{\mathcal{S}}(\delta + \delta_i)$$

Adversarial Patch

adversarial patch [Brown et al., 2017]

$$\min_{p} \mathbb{E}_{x,T} \ell(f, A(x, p, T), y_{\text{target}})$$

- p: an image patch
- T: a random transformation, *e.g.*, scaling followed by translation.
- $A(x, p, T)$: apply $T(p)$, a transformed patch, to an image x.
- Why do we need sampling over an image x and a transformation T?
- Do we need to constrain the perturbation size?

Attacker's Capability (=A Threat Model)

Whenever you meed an attack method, check out its assumption, called *attacker's capabilities* or a *threat model*.

1. an attacker has access to target model parameters.
   - FSGM, PGD, universal adversarial perturbation, adversarial patch
2. an attacker cannot add large perturbations, *i.e.*, $\|\delta\| \leq \varepsilon$.
   - FSGM, PGD, universal adversarial perturbation
3. an attacker has a capability to modify the input of a target model.
   - FSGM, PGD, universal adversarial perturbation
4. an attacker knows the true label of a target example x.
   - It is okay to use a model's prediction instead of the true label.

As can be seen, the adversarial patch works under weaker attacker's assumptions.

Outline

1. Introduction
2. White-box Attack Algorithms
3. Transfer Attack Algorithms
4. Black-box Attack Algorithms
5. Interpretation
6. Conclusion

Motivation on Transfer Attacks

key question

Can we attack if we don't have access to target model parameters?

- Often attackers don't have target model parameters, *e.g.*, Model as a Service (MaaS).

Usual Setup on Transfer Attacks

transfer attacks

1. find an adversarial perturbation δ from $\hat{y}$, *i.e.*, $\hat{y}(x+\delta) \neq \hat{y}(x)$
2. consider δ as an adversarial perturbation for $\hat{y}'$.

- $\hat{y}' : \mathcal{X} \to \mathcal{Y}$: a target model
- $\hat{y} : \mathcal{X} \to \mathcal{Y}$: a surrogate model
- x: a target example

(Probably) First Observation [Kurakin et al., 2017]

	source model	FGSM target model A	B	C	D	basic iter. target model A	B	C	D	iter l.l. target model A	B	C	D
top 1	A (v3)	100	56	58	47	100	46	45	33	100	13	13	9
	B (v3)	58	100	59	51	41	100	40	30	15	100	13	10
	C (v3 ELU)	56	58	100	52	44	44	100	32	12	11	100	9
	D (v4)	50	54	52	100	35	39	37	100	12	13	13	100
top 5	A (v3)	100	50	50	36	100	15	17	11	100	8	7	5
	B (v3)	51	100	50	37	16	100	14	10	7	100	5	4
	C (v3 ELU)	44	45	100	37	16	18	100	13	6	6	100	4
	D (v4)	42	38	46	100	11	15	15	100	6	6	6	100

- basic iter. = PGD
- the table is about error rates.
- observation: a perturbation from a weaker attack (*i.e.*, FGSM) is easier to transfer other models.

Outline

1. Introduction
2. White-box Attack Algorithms
3. Transfer Attack Algorithms
4. Black-box Attack Algorithms
5. Interpretation
6. Conclusion

Motivation

key question (again!)

Can we attack if we don't have access to target model parameters?

- transfer attacks are black-box attacks.

SimBA

SimBA [Guo et al., 2019]

$$\max_{\delta} \ell(f, x+\delta, y) \quad \text{subj. to} \quad \|\delta\| \leq \sqrt{T}\varepsilon \text{ and } \textit{queries} \leq B$$

Algorithm 1 SimBA in Pseudocode

procedure SIMBA($\mathbf{x}, y, Q, \epsilon$)
  $\delta = \mathbf{0}$
  $\mathbf{p} = p_h(y \mid \mathbf{x})$
  while $\mathbf{p}_y = \max_{y'} \mathbf{p}_{y'}$ **do**
    Pick randomly without replacement: $\mathbf{q} \in Q$
    for $\alpha \in \{\epsilon, -\epsilon\}$ **do**
      $\mathbf{p}' = p_h(y \mid \mathbf{x} + \delta + \alpha\mathbf{q})$
      if $\mathbf{p}'_y < \mathbf{p}_y$ **then**
        $\delta = \delta + \alpha\mathbf{q}$
        $\mathbf{p} = \mathbf{p}'$
        break
  return δ

- Q: a set of orthonormal bases
- T: the number of update
- The attack satisfies the query limit and has access to a predicted label distribution
- No backprop as it is a black box attack.
- Why $\sqrt{T}\varepsilon$?

$$\|\delta_T\|_2^2 = \|\sum_{t=1}^{T} \alpha_t \mathbf{q}_t\|_2^2 = \sum_{t=1}^{T} \|\alpha_t \mathbf{q}_t\|_2^2 = \sum_{t=1}^{T} \alpha_t^2 \|\mathbf{q}_t\|_2^2 \leq T\varepsilon^2$$

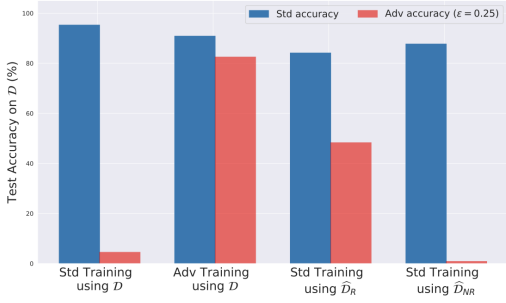
Outline

1. Introduction
2. White-box Attack Algorithms
3. Transfer Attack Algorithms
4. Black-box Attack Algorithms
5. Interpretation
6. Conclusion

Not Bugs But Features

claim [Ilyas et al., 2019]

Adversarial vulnerability is a direct result of our models' sensitivity to well-generalizing features in the data.



- deep models capture noisy patterns as features.
  - Interestingly, they do not capture shape information as features.
- noisy patterns are sensitive to some perturbations ($\approx$ adversarial perturbations)

Outline

1. Introduction
2. White-box Attack Algorithms
3. Transfer Attack Algorithms
4. Black-box Attack Algorithms
5. Interpretation
6. Conclusion

Conclusion

- We learn several white-box adversarial attack methods.
  - these basic methods could be building blocks for other attacks.
- Deep models captures noise patterns as features.
- How to learn a model to defend against adversarial perturbations?
- How does adversarial perturbations look like in other modalities, *e.g.*, text?

Reference I

T. B. Brown, D. Man?, A. Roy, M. Abadi, and J. Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

C. Guo, J. Gardner, Y. You, A. G. Wilson, and K. Weinberger. Simple black-box adversarial attacks. In *International conference on machine learning*, pages 2484–2493. PMLR, 2019.

A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry. Adversarial examples are not bugs, they are features. *Advances in neural information processing systems*, 32, 2019.

A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations (ICLR)*, 2017.

C. Lyu, K. Huang, and H.-N. Liang. A unified gradient regularization family for adversarial examples. In *IEEE International Conference on Data Mining (ICDM)*, pages 301–309. IEEE, 2015.

Reference II

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

A. Modas, S.-M. Moosavi-Dezfooli, and P. Frossard. Sparsefool: a few pixels make a big difference. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9087–9096, 2019.

S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.