Alon Rosen


# Adversarial Robustness in Classification

*I am not a witch!*[*]
*Tis but a scratch!*[†]

*Adversarial robustness* studies how a classifier's prediction behaves when its input is changed. In a *sensitivity* attack, a small label-preserving change alters the prediction; in an *invariance* attack, a change that should alter the label leaves the prediction essentially unchanged. Sensitivity is naturally captured by distance, while invariance is harder to formalize because deciding if two inputs are semantically different may require semantic judgment.

One might expect that a classifier that rarely errs should resist adversarial changes. Alas, even highly accurate classifiers might be vulnerable to sensitivity attacks.

The same *gradient-based* machinery used to train a classifier can also be used to attack it. During training, the input is fixed and the model parameters are changed to reduce the loss. In an attack, the parameters are fixed and the input is changed to increase it. Thus a method designed to improve classification can be repurposed to undermine the model's robustness. Interestingly, the very same attack can be used during training to immunize the model against it.

Such attack-specific protection leaves open whether other attacks exist. A *certificate* gives a stronger guarantee by ruling out all sensitivity attacks within a fixed neighborhood.

Cryptographic protection, when attainable, is different: even when harmful inputs may exist, no efficient attacker can find them.

The main points of this chapter are:

- In high dimension, a low misclassification rate can still leave errors near almost every input.
- Gradients can be used to mount sensitivity attacks; invariance attacks additionally require semantic information about whether the correct label has changed.
- Certified robustness can rule out all sensitivity attacks inside a specified local neighborhood.
- Existence and efficient discovery of harmful perturbations are different questions.
- Cryptographic guarantees, when available, are much stronger than protection against specific attacks: under stated assumptions, every efficient attacker in the model fails.

[*]Monty Python, *Monty Python and the Holy Grail*.
[†]Monty Python, *Monty Python and the Holy Grail*.

# 1 Sensitivity attacks

A *classification* problem assigns one of a fixed set of labels to each input. Examples include object recognition, spam filtering, medical diagnosis, and matching files to known prohibited content.

**Definition 1** (Classification problem)**.** *A classification problem consists of an input set $X$, a finite label set $Y = \{1, \ldots, K\}$, and a distribution $D$ over labeled examples $(x, y) \in X \times Y$. A classifier is a function $h : X \to Y$. Its classification error is*

$$\Pr_{(x,y)\sim D}[h(x) \neq y].$$

A *sensitivity* attack (also called an *adversarial example* or *evasion attack*) changes the prediction while leaving the correct label unchanged. For instance, Sir Bedevere might be fooled to think that the woman is a witch just because she wears a fake nose.

**Definition 2** (Sensitivity attack)**.** *Write $y(x)$ for the correct label of $x$, and let $\Delta(x) \subseteq X$ be the allowed perturbations of $x$. A sensitivity attack $A$ succeeds on a correctly classified input $x$ if $A(x) \in \Delta(x)$, $y(A(x)) = y(x)$, and $h(A(x)) \neq h(x)$. Its success rate is*

$$\Pr_{(x,y)\sim D}\big[A(x) \text{ succeeds} \mid h(x) = y\big].$$

The conditioning on $h(x) = y$ restricts attention to inputs classified correctly before the attack. Thus the success rate counts attack-induced failures, not pre-existing classification errors.

The set $\Delta(x)$ specifies the threat model. A common choice is the ball $\Delta(x) = \{x' : d(x, x') \leq \varepsilon\}$; smaller neighborhoods are more likely to preserve the correct label.

A key question is which metric to use. For images, $\ell_\infty$ is commonly used because it bounds the magnitude of the change at every pixel coordinate, while the $\ell_2$ norm controls the aggregate size of the perturbation, and an $\ell_0$-type constraint limits how many pixel coordinates may change.

The usual norms are related but not interchangeable. Since $\|v\|_\infty \leq \|v\|_2 \leq \sqrt{n}\,\|v\|_\infty$, an $\ell_2$ guarantee at radius $r$ gives an $\ell_\infty$ guarantee only at radius $r/\sqrt{n}$.

Ideally distance would follow the geometry of natural images, perhaps along an image manifold, but that geometry appears hard to capture explicitly. Short Euclidean distances are a rough approximation to short moves along such a manifold. Vision and multimodal models trained on natural images may capture some of this geometry implicitly.

# 2 Accuracy does not imply robustness

High accuracy means that a classifier makes few errors. We next demonstrate on the Hamming cube that, in high dimension, those errors can nevertheless lie close to almost every input.

**Theorem 1** (Nearby errors)**.** *For every fixed $\delta > 0$, any classifier on the Hamming cube with fixed positive error has an error within $O(\sqrt{n})$ bit flips of all but a $\delta$ fraction of inputs.*

*Proof.* Let $y(x)$ be the correct label of $x$. The error region

$$M = \{x : h(x) \neq y(x)\}$$

is the set of inputs that the classifier gets wrong.

Under the uniform distribution on $\{0,1\}^n$, the classification error is the fraction of the cube contained in $M$. Let $d_H(x, M)$ be the Hamming distance from $x$ to the nearest error, and define

$$M_d = \{x : d_H(x, M) \le d\}.$$

So $|M_d|/2^n$ is exactly the fraction of inputs that lie within $d$ bit flips of an error.

**Claim 1** (Concentration of error regions)**.** *If $\alpha = |M|/2^n > 0$, then for every fixed $0 < \delta < 1$ there is a constant $C_{\alpha,\delta} > 0$ such that, if $d \ge C_{\alpha,\delta}\sqrt{n}$, then $|M_d| > (1-\delta)2^n$.*

*Proof.* If $M$ itself were a Hamming ball, allowing $d$ additional bit flips would increase its radius by $d$, and already $d = \Theta(\sqrt{n})$ would suffice for the resulting ball to contain almost all strings.

The difficulty is that the error region $M$ of an arbitrary classifier can have any shape. To obtain a statement for every classifier, we therefore need to know which shape of $M$ minimizes the number of points lying within $d$ bit flips of the set. An inequality by Harper is precisely the right tool.

**Harper's inequality.** *Among sets $M$ of a given size, the number of points within $d$ bit flips from $M$ is minimized by a set $H$ shaped like a Hamming ball, plus possibly part of its outermost layer.*

Let $H$ be the guaranteed set, and let $H_d$ denote the points within $d$ bit flips of $H$. Then

$$|M_d| \ge |H_d|.$$

For some integer $r$, $H$ contains every string at Hamming distance less than $r$ from its center, possibly some strings at distance $r$, and no string at larger distance. Consequently, $H_d$ contains every string at distance less than $r + d$ from the center of $H$.

By Hoeffding, the Hamming distance of a uniformly random string from the center $c$ of $H$ is concentrated around $n/2$. Specifically, for every $t > 0$,

$$\Pr_{x\in\{0,1\}^n}\left[\left|d_H(x,c) - \frac{n}{2}\right| \ge t\right] \le 2e^{-2t^2/n}.$$

Since $H$ occupies an $\alpha$ fraction of the cube, its radius $r$ must therefore be at least $n/2 - O_\alpha(\sqrt{n})$. Likewise, all but a $\delta$ fraction of the cube lies within distance $n/2 + O_\delta(\sqrt{n})$ of the center of $H$.

Therefore, if $d \ge C_{\alpha,\delta}\sqrt{n}$ for large enough $C_{\alpha,\delta}$, then

$$r + d \ge \frac{n}{2} - O_\alpha(\sqrt{n}) + C_{\alpha,\delta}\sqrt{n} \ge \frac{n}{2} + O_\delta(\sqrt{n}).$$

Hence $H_d$ contains more than a $1-\delta$ fraction of the cube, and so must $M_d$. □

Since $M$ is the classifier's error region, the claim proves the theorem. □

Analogous bounds hold for other product distributions. Under a standard Gaussian on $\mathbb{R}^n$, any error region of fixed positive probability lies within constant $\ell_2$ distance of almost every input. But a nearby error need not be a sensitivity attack: it may change the true label or lie outside $\Delta(x)$.

How does Hamming distance translate to sensitivity attacks in various norms? For bit vectors, changing $d$ coordinates gives $\ell_0$ and $\ell_1$ distance $d$, $\ell_2$ distance $\sqrt{d}$, and $\ell_\infty$ distance 1.

For $x \in \{0,1\}^n$, every coordinate flip has unit Hamming cost. Over $\mathbb{R}^n$, perturbations have numerical magnitude: in fixed precision, high-order bits move coordinates farther than low-order bits. Gradient-based attacks search directly for small changes in these real-valued coordinates.

# 3 Attacks and defenses

Theorem 1 says only that a nearby error exists, not how to find one efficiently. A *gradient-based* attacker searches for one by following the gradient of the loss with respect to the input $x \in \mathbb{R}^n$, changing it in a direction that increases the loss. This exploits the training procedure, which instead follows the gradient of the same loss with respect to the model parameters $\theta \in \mathbb{R}^p$.

## 3.1 Gradient-based attacks

Suppose the classifier assigns a score $F_{\theta,y}(x)$ to each label $y$ and predicts

$$h_\theta(x) = \arg\max_{y \in Y} F_{\theta,y}(x).$$

These scores define a real-valued loss $L(\theta, x, y)$, small when the classifier fits the labeled example well. If the loss is differentiable in $x$, then for a small perturbation $\delta$,

$$L(\theta, x + \delta, y) \approx L(\theta, x, y) + \nabla_x L(\theta, x, y)^\top \delta.$$

The gradient $\nabla_x L$ gives a direction for changing the input, while the norm and radius specify $\Delta(x)$. Common choices are $\ell_\infty$ and $\ell_2$.

With *white-box* access, gradients of the loss can guide the attack; with *black-box* access, one may instead estimate them from queries or transfer an attack from another model. Attack success is estimated on separate labeled *hold-out* examples not used for training.

## 3.2 Adversarial training

In *adversarial training*, one turns this attack back on itself, using it in training to make the classifier more resistant to it by optimizing $\theta$ against the worst allowed perturbation:

$$\min_\theta \ \mathbb{E}_{(x,y)\sim D} \left[ \max_{\delta \in \Delta(x)} L(\theta, x + \delta, y) \right].$$

The inner maximization searches for a perturbation that raises the loss; the outer minimization updates $\theta$ to lower the resulting worst-case loss. In practice, the inner problem is approximated by gradient-based attacks of this kind, and $\Delta(x)$ should contain only label-preserving changes.

Robustness on training examples need not extend to fresh examples. In some settings, robust generalization requires allowing the learner to leave the original classifier family, for example by outputting a majority vote of its members.

At standard perturbation budgets, gradient sensitivity attacks can reach about 90% success against standard vision models; adversarial training cuts this sharply, to roughly 25% on ImageNet at $\ell_\infty$ radius $4/255$ for leading models.

## 3.3 Certified robustness

A key issue is that Definition 2 asks only whether an attack succeeds on a typical input. However, in some settings even a single successful attack can have disastrous consequences, for example if it causes one stop sign to be misclassified.

*Certified robustness* addresses this question by replacing attack-based testing with a strong guarantee: for a given input, a successful certificate establishes that no perturbation in a specified neighborhood can change the prediction.

**Definition 3** (Robustness certificate)**.** *Fix a classifier $h : X \to Y$ and a perturbation rule $\Delta$ with $\Delta(x) \subseteq X$. An efficiently computable predicate $C_{h,\Delta} : X \to \{0, 1\}$ is a robustness certificate if $C_{h,\Delta}(x) = 1$ implies $h(x') = h(x)$ for every $x' \in \Delta(x)$.*

If $C_{h,\Delta}(x) = 1$, the prediction is guaranteed to remain unchanged under every allowed perturbation of $x$. If $C_{h,\Delta}(x) = 0$, no conclusion follows: the prediction may still be unchanged throughout $\Delta(x)$, or some allowed perturbation may change it.

To address sensitivity, adversarial training and certification should use perturbations that preserve the true label. Closeness alone does not ensure label preservation, since the nearby errors guaranteed by concentration may have different true labels. For a certificate to be useful, it should accept many correctly classified inputs.

One method for achieving certified robustness is *randomized smoothing*. Replace $h(x)$ by the label most likely under a Gaussian perturbation of $x$, estimated from many noisy copies. A sufficiently large winning margin certifies that this label cannot change within an $\ell_2$ ball. For fixed voting probabilities, the certified $\ell_2$ radius is proportional to the Gaussian noise scale $\sigma$; a certified $\ell_2$ radius $r$ also guarantees an $\ell_\infty$ radius $r/\sqrt{n}$. The certificate holds with a stated confidence level.

Another method is *bound propagation*. Propagate guaranteed neuron ranges under all allowed perturbations, yielding worst-case output-score bounds. If one label stays above all others, the prediction is certified; varying $\varepsilon$ gives the largest certifiable $\ell_\infty$ radius for that input. Such radii are typically small, but each coordinate may change by up to $\varepsilon$. The certifiable radius also depends on the classifier and data, so dimension $n$ alone gives no positive lower bound.

### 3.4 Cryptographic robustness

A natural cryptographic idea is to restrict the attack $A$ to run in time at most $t$, and require every such attack to have success rate smaller than $\epsilon$. In the context of certified robustness:

- ***Computational certification:*** for every time-$t$ attacker $A$,

$$\Pr_{(x,y)\sim D}\big[C_{h,\Delta}(x) = 1 \ \wedge \ A(x) \text{ succeeds} \mid h(x) = y\big] \leq \epsilon.$$

Here $C$ may certify an input even when a successful sensitivity perturbation exists in $\Delta(x)$. This relaxation may permit a larger $\Delta(x)$, or acceptance of a larger portion of the data distribution.

Whether a computational guarantee can be realized in a classification setting remains open. Cryptography has the potential to make harmful perturbations hard to find or at the very least enlarge the radius of certification, not only protecting against gradient attacks but even attacks that are yet to be conceived. Realizing this vision is a fascinating challenge.

One simple idea is to authenticate the input. Let $h_0$ be an accurate classifier. Before an attacker can modify $x$, a trusted party computes a signature $\sigma = \mathrm{Sign}_{sk}(x)$. Given $(x, \sigma)$, the classifier rejects if the signature is invalid, and otherwise returns $h_0(x)$. Indeed, changing an accepted pair $(x, \sigma)$ into an accepted pair $(x', \sigma')$ with $x' \neq x$ forges a signature on the new message $x'$.

**Theorem 2.** *Assuming secure signatures, there is a computationally robust classifier achieving the best attainable classification error.*

This protects only modifications made after authentication. If a scene is changed before a trusted sensor signs it, the altered input is honestly authenticated. The construction adds a signature and rejection, so it has the undesirable property of changing $h_0$'s interface.

# 4 Invariance attacks

In a sensitivity attack, the correct label stays fixed but the classifier's prediction changes. An *invariance* attack reverses these conditions: the correct label changes while the prediction stays fixed. Just like the Black Knight who refuses to surrender in spite of the massive limb loss. We do not impose the same locality restriction on invariance attacks.

**Definition 4** (Invariance attack)**.** *Write $y(x)$ for the correct label of $x$. An invariance attack $A$ is successful on a correctly classified input $x$ if $y(A(x)) \neq y(x)$ and $h(A(x)) = h(x)$.*

How might one mount such an attack? A trained network may map $x$ to a feature representation $E_\theta(x)$. In the Sir Bedevere example, its coordinates might encode the fake nose and other cues separating witches from non-witches. Starting from a non-witch $x$ and a witch $z$, one can search near $z$ for an input whose representation moves toward $E_\theta(x)$, for example by minimizing

$$J(u) = \|E_\theta(u) - E_\theta(x)\|_2^2 + \lambda\|u - z\|_2^2.$$

Gradient-based search may find a pair $(x, u)$ whose feature representations are highly contracted in $\ell_2$, but contraction alone does not make an invariance attack. One must also verify that $x$ and $u$ have different correct labels while the classifier gives them the same prediction; maintaining this distinction throughout the search requires checking the labels along the path.

## 4.1 Random projections

The attack above keeps $u$ close to $z$ in input $\ell_2$ distance, and hence far from $x$ when $x$ and $z$ are well separated, while driving $E_\theta(u)$ close to $E_\theta(x)$ in feature $\ell_2$ distance. This suggests the geometric question: can a random projection prevent such contractions?

Let $A \in \mathbb{R}^{m\times n}$ be a suitably normalized Gaussian matrix, and consider the linear map

$$A : \mathbb{R}^n \to \mathbb{R}^m, \qquad A(x) = Ax.$$

**Lemma 1** (Johnson–Lindenstrauss)**.** *For any fixed finite set $S \subset \mathbb{R}^n$ chosen independently of $A$, if $m = O(\varepsilon^{-2} \log |S|)$, then with high probability $\|A(x) - A(x')\|_2 \in (1 \pm \varepsilon)\|x - x'\|_2$ for every $x, x' \in S$.*

Thus a pair that is far in input $\ell_2$ distance cannot become much closer after projection. This is exactly the contraction targeted by the attack above.

However, an adaptive attacker sees $A$ *before* choosing the pair. Since $m < n$, linear algebra finds a nonzero $v \in \ker(A)$, so $A(x) = A(x + v)$ for every $x$. Scaling $v$ gives arbitrarily distant pairs with identical features. Thus $A$ cannot prevent adaptive $\ell_2$ contractions.

## 4.2 Adaptively-robust projections

To seek robustness against adaptive attacks, discretize and bound the input $x$ to be in $([-B, B] \cap \mathbb{Z})^n$ and round each output coordinate to a grid of width $\varepsilon$ to obtain the map:

$$H_A(x) = \lfloor Ax \rfloor_\varepsilon.$$

Finding a strongly contracting pair $x \neq x'$ is the same as finding a nonzero bounded integer vector $\mathbf{v} = x - x'$ whose Gaussian projection is unusually small.

**Definition 5** (Contracting hypergrid vector (CHV) problem)**.** *Given $A \in \mathbb{R}^{m\times n}$ with independent $\mathcal{N}(0,1)$ entries, find a nonzero $\mathbf{v} \in ([-B, B] \cap \mathbb{Z})^n$ such that*

$$\|A\mathbf{v}\|_2 < \kappa\sqrt{m}\,\|\mathbf{v}\|_2.$$

For any fixed nonzero $\mathbf{v}$, $\|A\mathbf{v}\|_2$ is typically on the order of $\sqrt{m}\,\|\mathbf{v}\|_2$. CHV asks whether an efficient algorithm, after seeing $A$, can find a bounded integer vector that contracts much more.

**Theorem 3** (CHV hardness from CLWE)**.** *A polynomial time algorithm for CHV gives a polynomial time algorithm for CLWE.*

*Proof.* A CLWE instance is $(A, \mathbf{b})$, where the rows of $A \in \mathbb{R}^{m\times n}$ are the $m$ sample vectors and

$$\mathbf{b} = \gamma A\mathbf{w} + \mathbf{e} \pmod 1,$$

for a hidden unit vector $\mathbf{w} \in \mathbb{R}^n$ and a noise vector $\mathbf{e} \in \mathbb{R}^m$ with independent Gaussian coordinates of width $\beta$; the null distribution instead chooses $\mathbf{b}$ uniformly in $[0,1)^m$, independently of $A$.

To match the CLWE notation, we instantiate CHV with a matrix in $\mathbb{R}^{n\times m}$ rather than $\mathbb{R}^{m\times n}$. Run the CHV algorithm on $A^\top \in \mathbb{R}^{n\times m}$ to obtain a nonzero $\mathbf{v} \in ([-B, B] \cap \mathbb{Z})^m$ satisfying

$$\|A^\top\mathbf{v}\|_2 < \kappa\sqrt{n}\,\|\mathbf{v}\|_2.$$

Accept if $|\mathbf{b}^\top\mathbf{v} \bmod 1| < 1/4$. Under the null distribution, $\mathbf{v}$ depends only on $A$, so $\mathbf{b}^\top\mathbf{v} \bmod 1$ is uniform and the test accepts with probability $1/2$. For a CLWE instance,

$$\mathbf{b}^\top\mathbf{v} = \gamma\mathbf{w}^\top A^\top\mathbf{v} + \mathbf{e}^\top\mathbf{v} \pmod 1.$$

The CHV guarantee bounds the first term by $\gamma\kappa\sqrt{n}\,\|\mathbf{v}\|_2$. Conditioned on $A$ and $\mathbf{v}$, the second is Gaussian of width $\beta\|\mathbf{v}\|_2$. For appropriate parameters, the first term is smaller than $1/8$ and the second has magnitude below $1/8$ with probability at least $3/4$. Hence the test accepts CLWE with probability at least $3/4$, giving constant advantage over the null distribution. □

A strongly contracting pair for $H_A$ yields a bounded integer difference vector solving CHV. Thus, for appropriate parameters, $H_A$ is adaptively robust against large $\ell_2$ contractions.

### 4.3 Towards robust classification

By Theorem 3, $H_A$ blocks the geometric route used by the gradient attack above: an efficient attacker cannot find a pair that is far in input $\ell_2$ distance but anomalously close after projection. An adaptive invariance attacker must therefore seek another route.

Random projections are useful feature extractors. This suggests sampling $A$ once, freezing it, and training only a classifier $g$ on the projected features:

$$h(x) = g(H_A(x)).$$

The robust projection protects this feature-extraction stage against large adaptive contractions, but does not rule out semantically different inputs that fall in the same decision region of $g$.

A possible next step is to first train an embedding $E_\theta$ that maps semantic similarity to Euclidean proximity and semantic difference to Euclidean separation. The embedding may be expanding, making this geometric training task potentially easier by leaving ample room in a high-dimensional space to separate different semantic equivalence classes. After quantization, $H_A$ can then protect this learned geometry against large adaptive contractions. Turning the composition into an end-to-end robust classifier remains open.

## 5 Historical notes

The modern study of sensitivity attacks began with Szegedy et al. [14]. Goodfellow et al. introduced the fast sign attack, while Madry et al. systematized projected-gradient attacks and adversarial training [7, 12]. The Hamming-cube phenomenon was studied by Diochnos et al. using Harper's inequality [4, 9]; Fawzi et al. gave related Gaussian analogues without label preservation [5].

Montasser et al. showed that some families of classifiers can be robustly learned by taking a majority vote of their members, but not by selecting a single member [13]. Gowal et al. developed interval bounds; Cohen et al., randomized smoothing [8, 2]. Degwekar et al. and Garg et al. studied computational hardness [3, 6]. Invariance attacks are due to Jacobsen et al. [10], random projections to Johnson and Lindenstrauss [11], and CHV/rounded projections to Bogdanov et al. [1].

## References


[1] A. Bogdanov, A. Rosen, N. Vafa, and V. Vaikuntanathan. *Adaptive Robustness of Hypergrid Johnson–Lindenstrauss*. STOC, 2026.

[2] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter. *Certified Adversarial Robustness via Randomized Smoothing*. ICML, 2019.

[3] A. Degwekar, P. Nakkiran, and V. Vaikuntanathan. *Computational Limitations in Robust Classification and Win-Win Results*. arXiv:1902.01086, 2019.

[4] D. I. Diochnos, S. Mahloujifar, and M. Mahmoody. *Adversarial Risk and Robustness: General Definitions and Implications for the Uniform Distribution*. NeurIPS, 2018.

[5] A. Fawzi, H. Fawzi, and O. Fawzi. *Adversarial Vulnerability for Any Classifier*. NeurIPS, 2018.

[6] S. Garg, S. Jha, S. Mahloujifar, and M. Mahmoody. *Adversarially Robust Learning Could Leverage Computational Hardness*. arXiv:1905.11564, 2019.

[7] I. J. Goodfellow, J. Shlens, and C. Szegedy. *Explaining and Harnessing Adversarial Examples*. ICLR, 2015.

[8] S. Gowal et al. *On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models*. arXiv:1810.12715, 2018.

[9] L. H. Harper. *Optimal Numberings and Isoperimetric Problems on Graphs*. Journal of Combinatorial Theory, 1(3):385–393, 1966.

[10] J.-H. Jacobsen, J. Behrmann, R. Zemel, and M. Bethge. *Excessive Invariance Causes Adversarial Vulnerability*. ICLR, 2019.

[11] W. B. Johnson and J. Lindenstrauss. *Extensions of Lipschitz Mappings into a Hilbert Space*. Contemporary Mathematics, 26:189–206, 1984.

[12] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. *Towards Deep Learning Models Resistant to Adversarial Attacks*. ICLR, 2018.

[13] O. Montasser, S. Hanneke, and N. Srebro. *VC Classes are Adversarially Robustly Learnable, but Only Improperly*. COLT, 2019.

[14] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. *Intriguing Properties of Neural Networks*. ICLR, 2014.