Fabian Herzog

# Fooling Neural Networks

# Machine Learning and Neural Networks

- Neural Networks are very successful function approximation machines
- Benchmark results in fields of image processing, computer vision, speech recognition, …
- Deep Learning lately became a major buzzword

# Machine Learning Classifier

Example: Image Recognition

Given an input image, the classifier should answer with a probability distribution of output classes.

Classifier

$$\begin{bmatrix} 0.12 \\ 0.64 \\ \vdots \\ 0.08 \end{bmatrix}$$

# Machine Learning Classifier

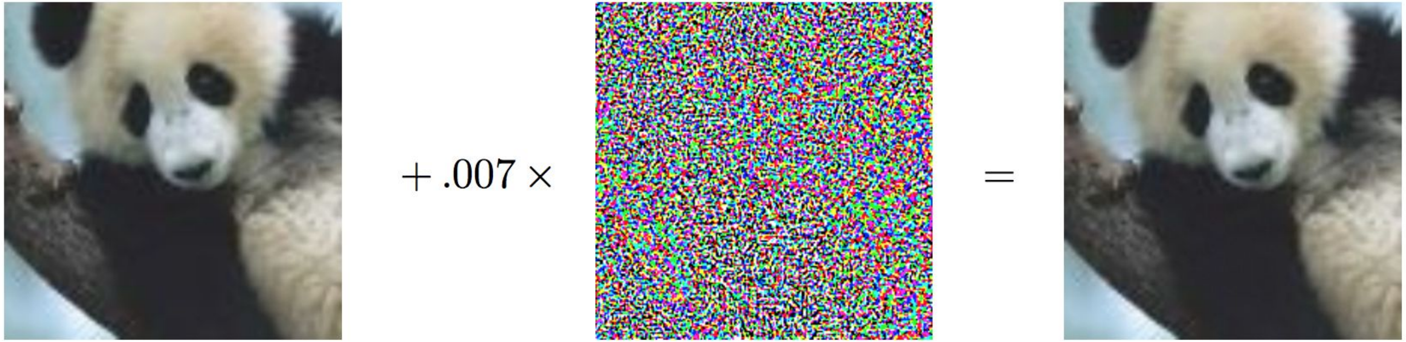
Example: Image Recognition

Given an input image, the classifier should answer with a probability distribution of output classes.

$$\longrightarrow \boxed{\textbf{Classifier}} \longrightarrow \begin{bmatrix} 0.12 \\ 0.64 \\ \vdots \\ 0.08 \end{bmatrix}$$

# Adversarial Example

A small perturbation in the input image can lead to misclassification:




| | | |
|---|---|---|
| $\boldsymbol{x}$ | $\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$ | $\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$ |
| "panda" | "nematode" | "gibbon" |
| 57.7% confidence | 8.2% confidence | 99.3 % confidence |

# More Adversarial Examples


Lionfish


Tiger Beetle


Magpie


Rhodesian Ridgeback

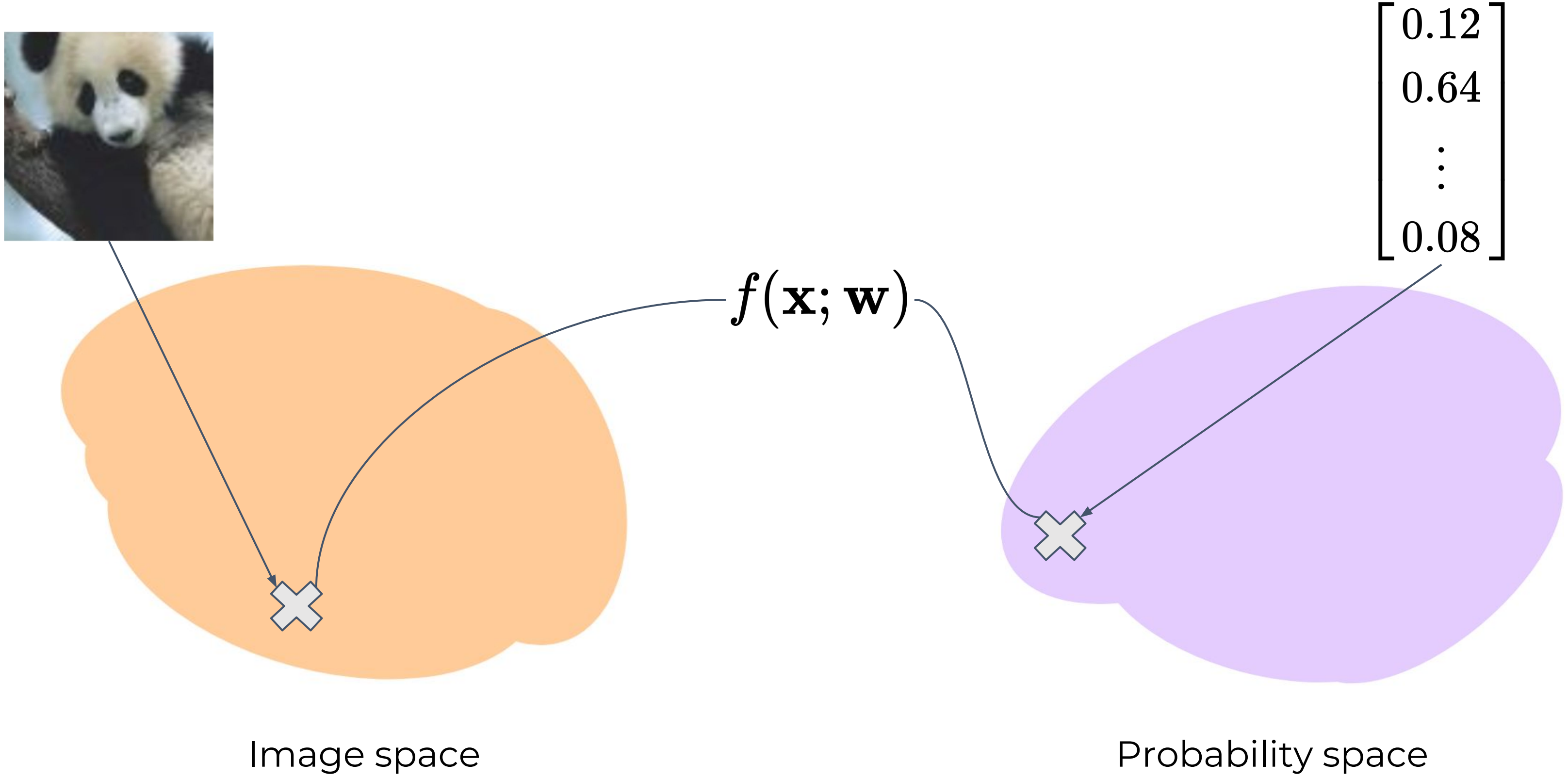

Eggnog


Green Mamba

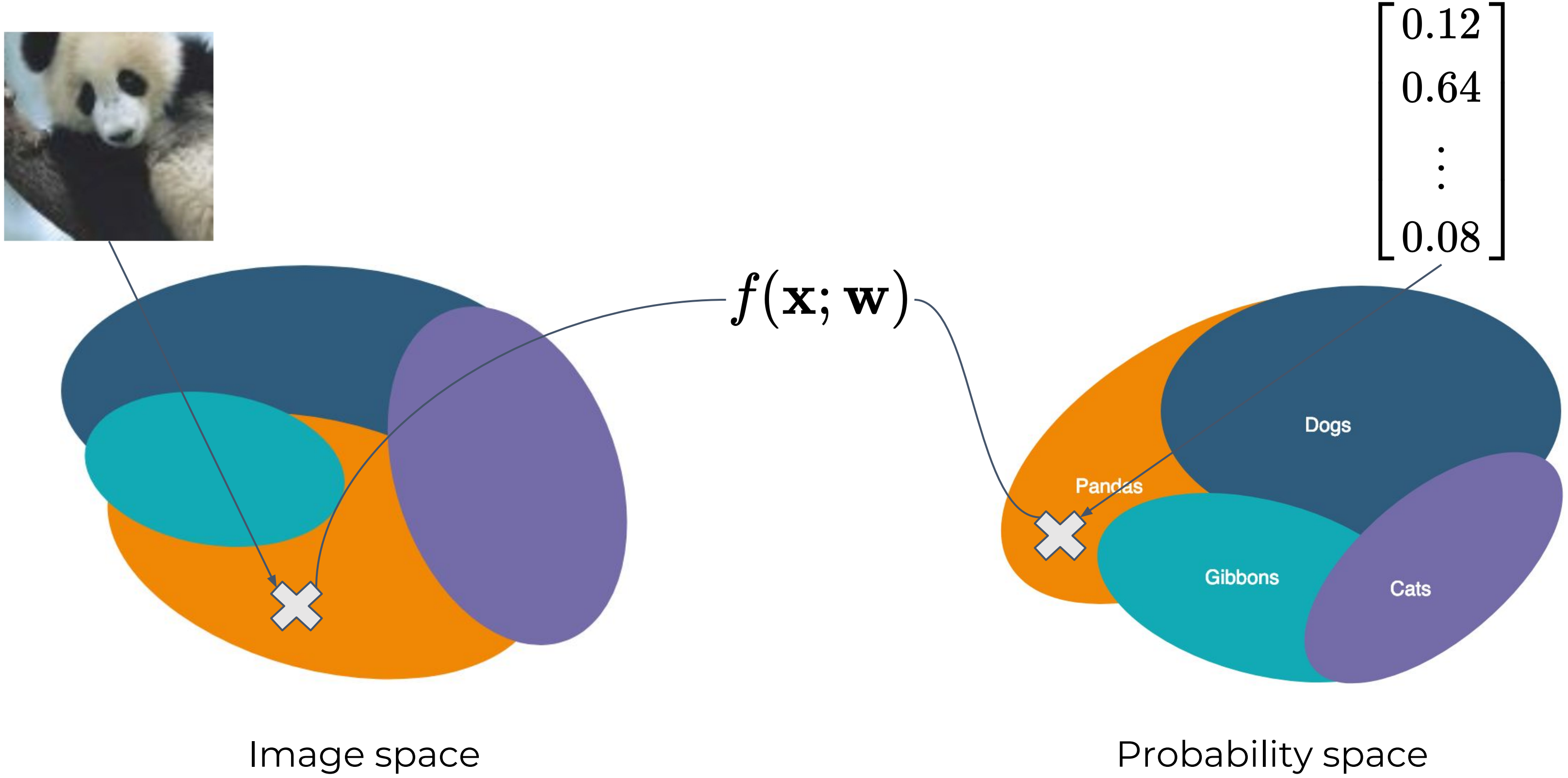

Great Pyrenees


Shopping Basket

# What does happen here!?



# What does happen here!?



# What does happen here!?

Adjust the input image slightly in the direction that leads to a misclassification

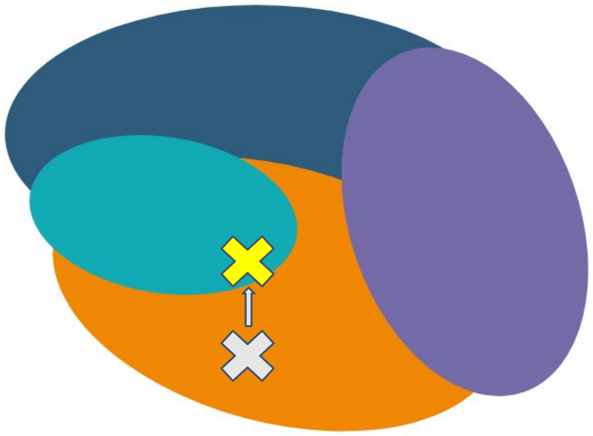
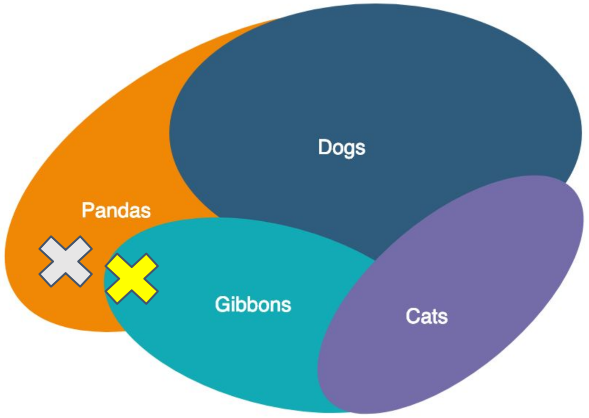
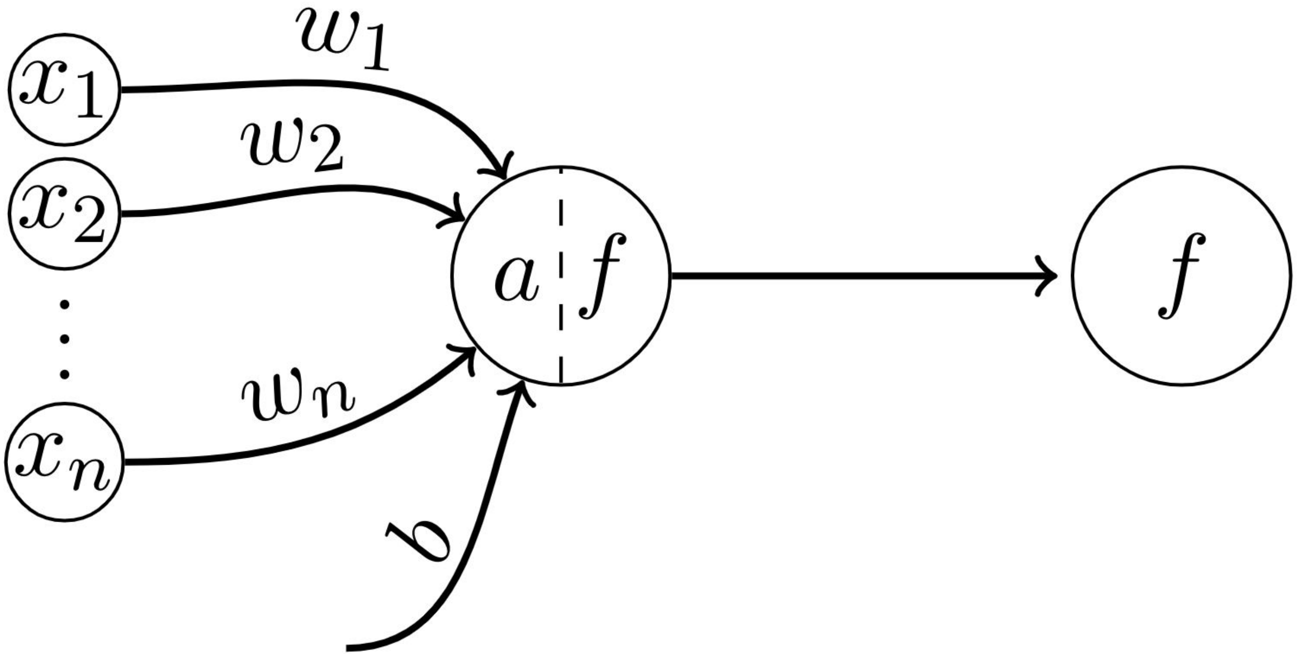


Image space



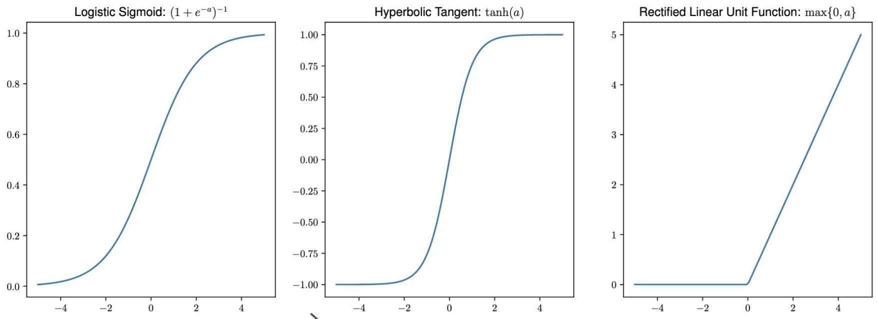

Probability space

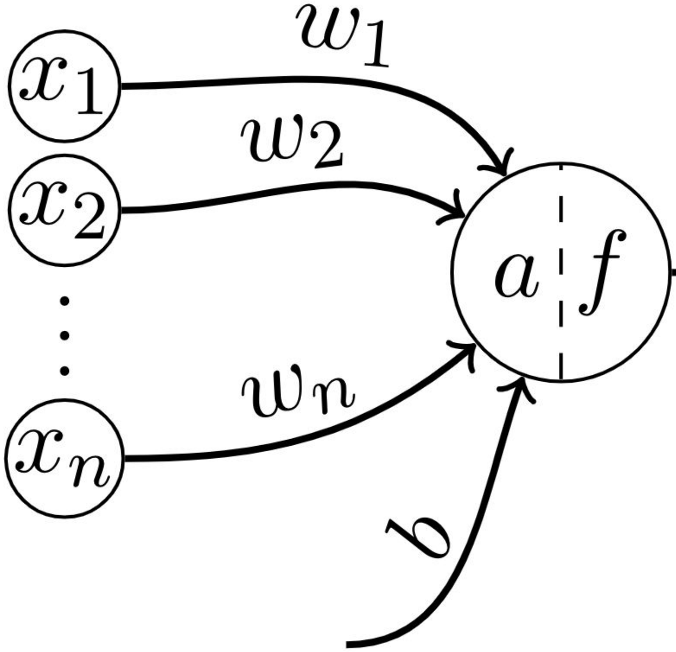
# How does it work?

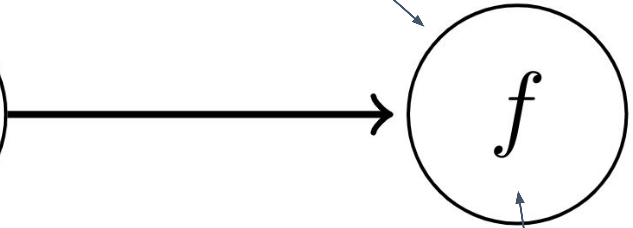



Single Neuron Classifier

# How does it work?




Activation

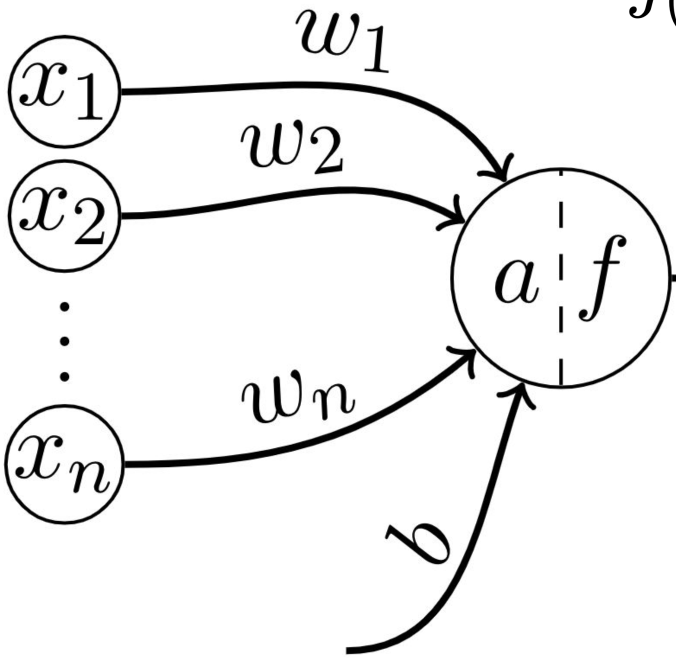
$$a = \sum_i x_i w_i + b = \langle \mathbf{x}, \mathbf{w} \rangle + b$$

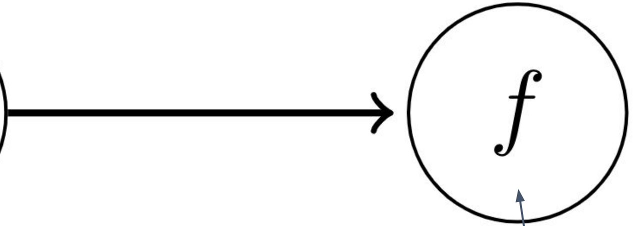
Single Neuron Classifier

# How does it work?

Minimize loss

$$J(\mathbf{w}, b; \mathbf{x}, y)$$




Activation

$$a = \sum_i x_i w_i + b = \langle \mathbf{x}, \mathbf{w} \rangle + b$$

Single Neuron Classifier

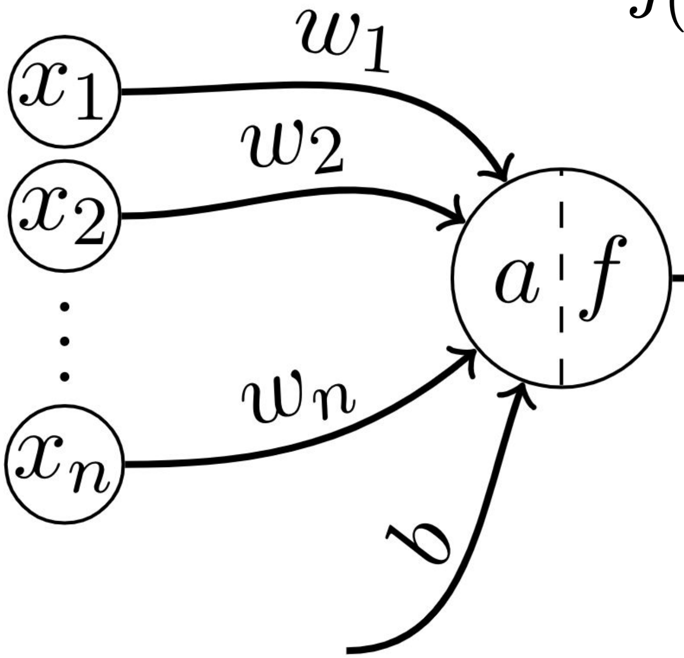
# How does it work?

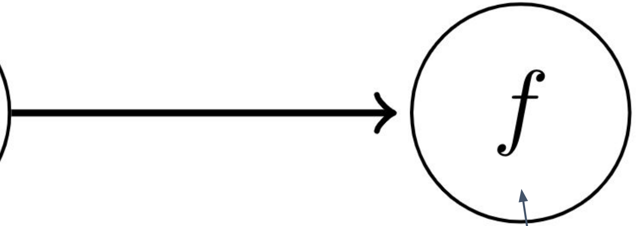
Minimize loss

$$J(\mathbf{w}, b; \mathbf{x}, y)$$

Move in the opposite direction of

$$\nabla_{\mathbf{w}} J(\mathbf{w}, b; \mathbf{x}, y)$$




Activation

$$a = \sum_i x_i w_i + b = \langle \mathbf{x}, \mathbf{w} \rangle + b$$

Single Neuron Classifier

# Now we do the opposite!

- Instead of following $\nabla_{\mathbf{w}} J(\mathbf{w}, b; \mathbf{x}, y)$ , we follow

$$\nabla_{\mathbf{x}} J(\mathbf{w}, b; \mathbf{x}, y)$$

until we achieve misclassification

# This is what happens:

We follow $\nabla_{\mathbf{x}} J(\mathbf{w}, b; \mathbf{x}, y)$ and move our point in image space, leading to a probability vector corresponding to another class.

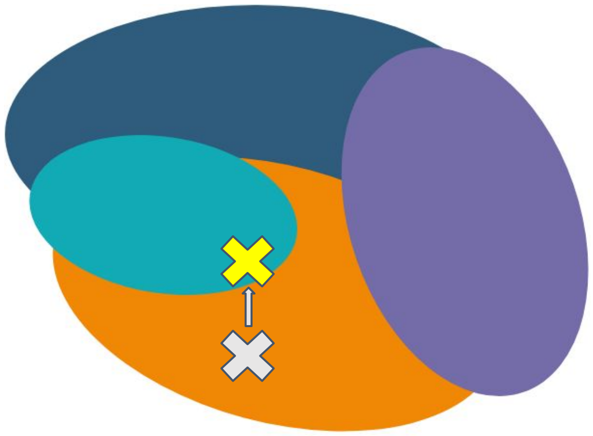
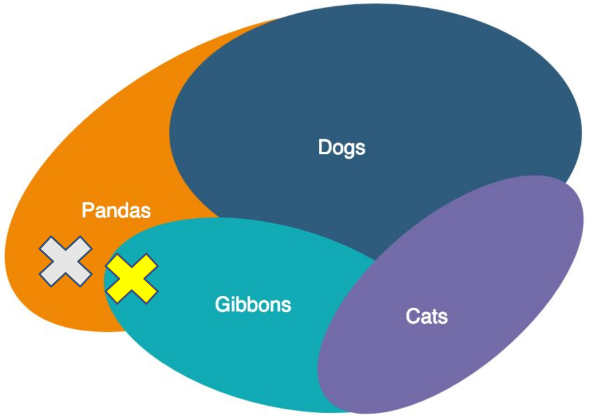


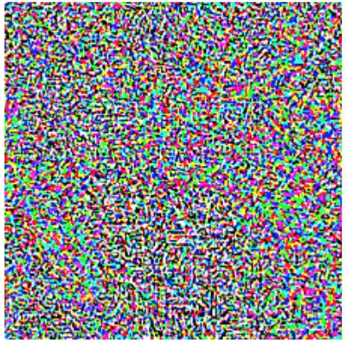
Image space




Probability space

# Adversarial Example

$+ .007 \times$



$=$

$\boldsymbol{x}$

"panda"

57.7% confidence

$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"

8.2% confidence

$\boldsymbol{x} + \epsilon \text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

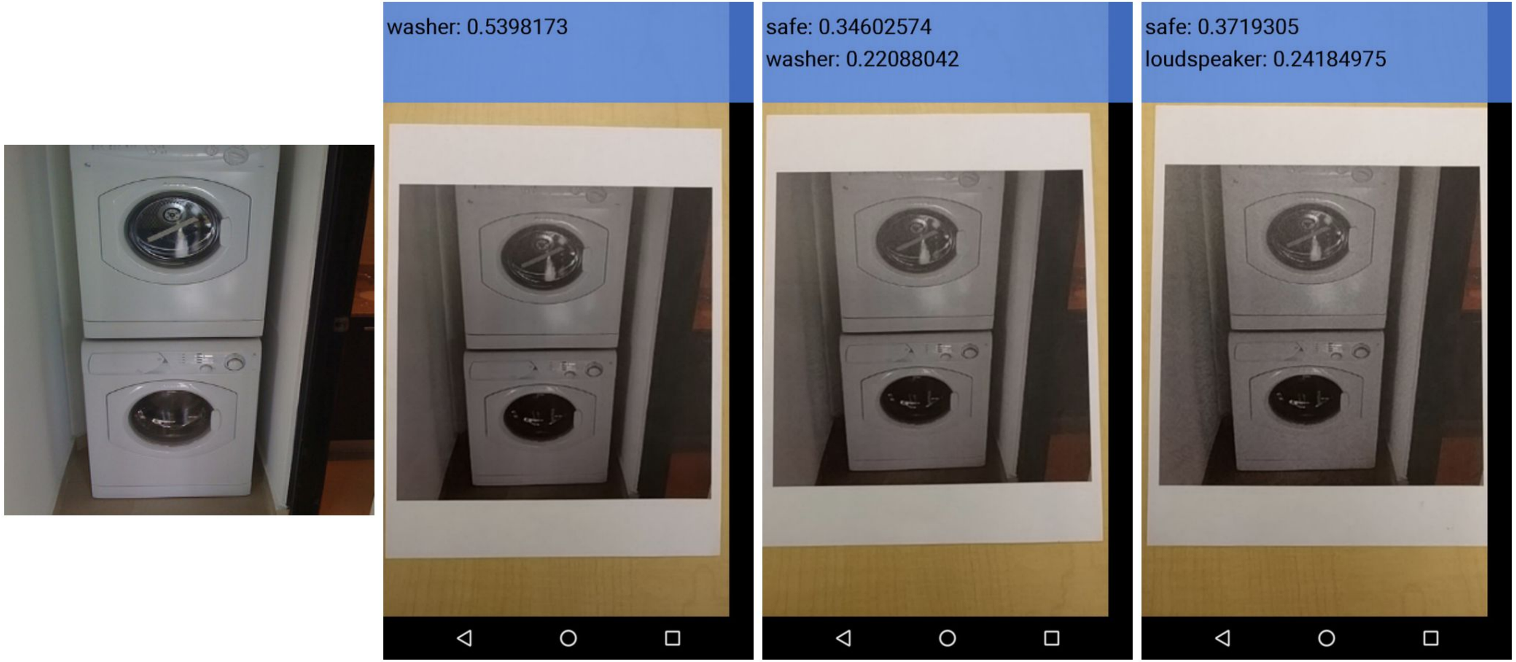
"gibbon"

99.3 % confidence

# Problems

- A model is easy to train if it’s more or less linear
- The more linear a model is, the easier the optimization
- But the same linearity can be used against the network
- LSTMs, ReLUs, ...
- Small perturbations can lead to misclassifications of the ImageNet database for more than 99% of the images (there are 14 Million classified images in ImageNet)

# Real World Problems




(a) Image from dataset (b) Clean image (c) Adv. image, $\epsilon = 4$ (d) Adv. image, $\epsilon = 8$

# Potential Targets

- Robots / Self-driving cars?
- Financial data (manipulate credit rating)?
- Other areas where people rely on the rationality of algorithms

# Types of Attacks

Original image: $\mathbf{x}$

Adversarial Image: $\tilde{\mathbf{x}}$

Set of classes: $C := \{y_1, \ldots, y_m\}$

True label: $y^* \in C$

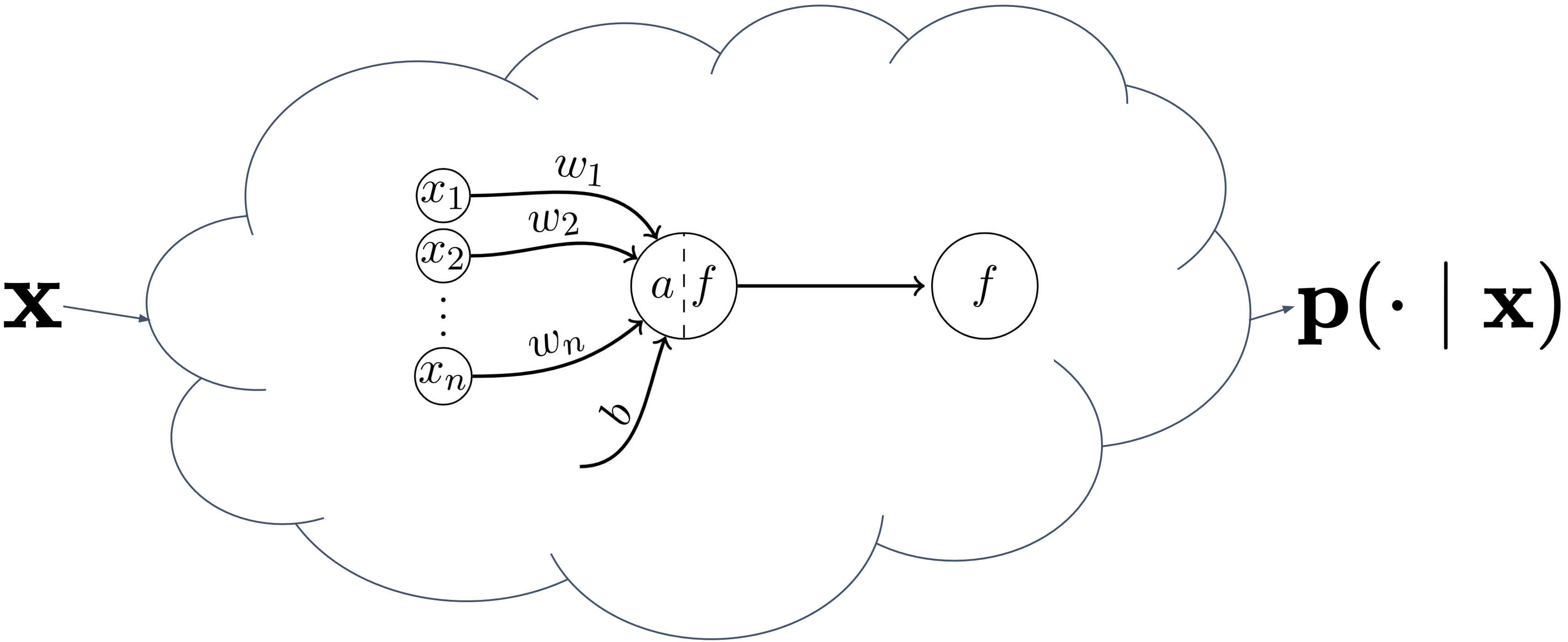
- **Non-targeted attack:**

$$\text{argmax}_{y \in C}\, p(y \mid \tilde{\mathbf{x}}) \neq y^*$$

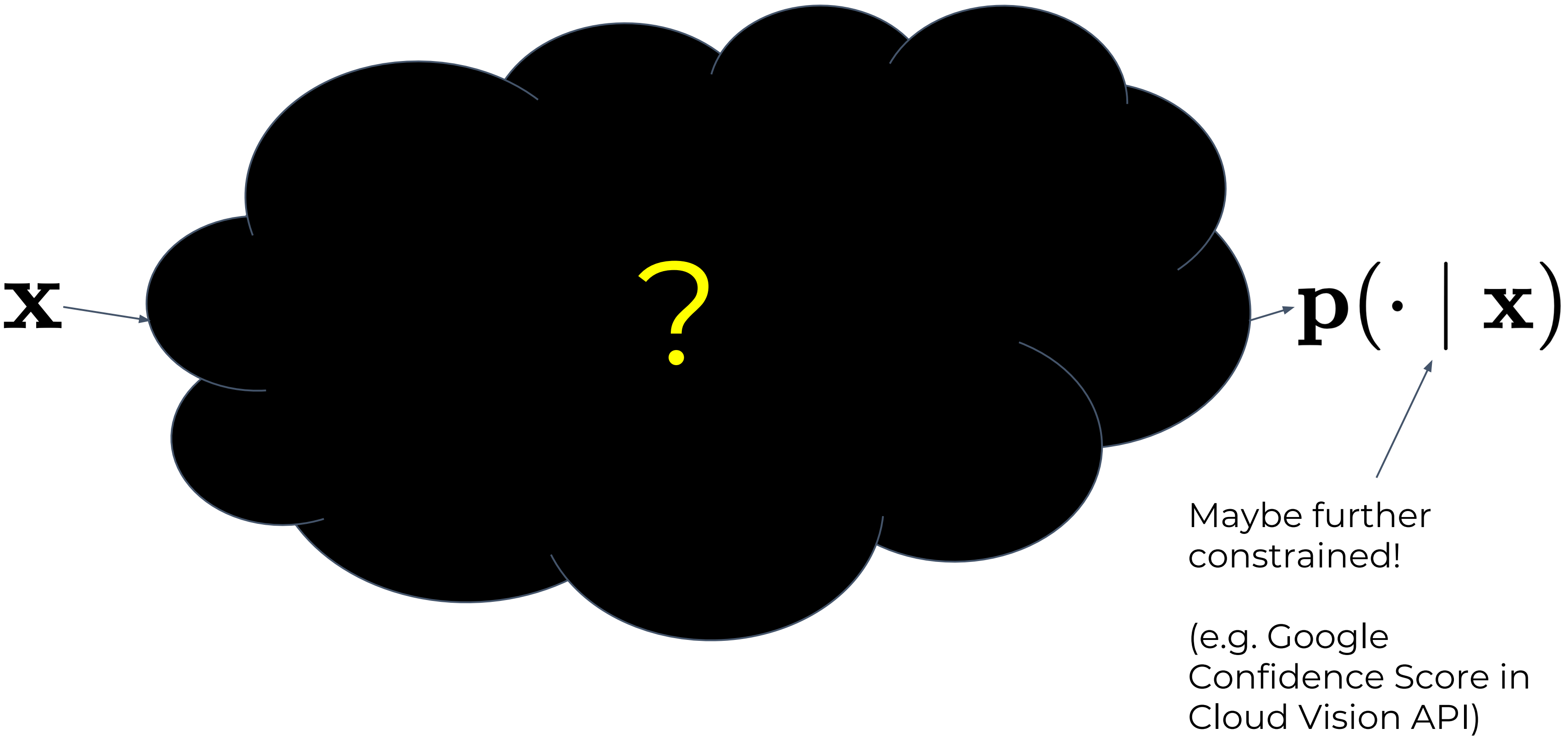
- **Targeted attack:**

$$\text{argmax}_{y \in C}\, p(y \mid \tilde{\mathbf{x}}) = y_t$$

# Types of Attacks: Whitebox



# Types of Attacks: Blackbox



# Counter Measures

## Blackbox:

- We hide the weights, the gradient, the loss function and everything else and only output the probability vector for the given input!
- **This is not sufficient!**

# Blackbox as a Counter Measure

- The attacker can train his/her own network based on the input/output relations of the original network
- Empirically, adversarial inputs for one network tend to be adversarial for another network
- Even if we limit the number of queries, the blackbox is not a sufficient counter measure (cf. Ilyas 2018)
- Even if we limit the output, the blackbox is not a sufficient counter measure (cf. Ilyas 2018)

## Counter Measures

- Most attacks use the neural network gradient
- We find a search direction by modifying the gradient and move in a promising direction
- Hiding the gradient (like in the blackbox case) does not make the model more robust (cf. Goodfellow 2017)

# Some Counter Measures

## Adversarial Training

- While training, generate adversarial examples yourself and train the network to correctly classify the perturbated examples!
- Generation of examples is fast (using the method presented in these slides)
- However, there is a large number of possible perturbations...

# Some Counter Measures

**Defensive distillation:**

- Use the first trained network (with harder decision boundaries) to train a second, smoothed network (with smoothed decision boundary)

# What to do / Further Reading?

- Cleverhans Library for generating adversary examples and testing models (http://www.cleverhans.io/)
- We currently do not really know “what to do”. There is no perfect solution
- There is no true theoretical understanding / no underlying mathematical theory that would allow us to derive a solid and adaptive counter measure
- For now, we can only try to make ML models more robust
- **Arms race...**
- Maybe there is no solution after all

# Scientific Sources / Reading List


Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." *arXiv preprint arXiv:1412.6572* (2014).

Ilyas, Andrew, et al. "Black-box adversarial attacks with limited queries and information." *arXiv preprint arXiv:1804.08598*(2018).

Kurakin, Alexey, Ian Goodfellow, and Samy Bengio. "Adversarial examples in the physical world." *arXiv preprint arXiv:1607.02533* (2016).