Recent Advances in Responsible AI
Robust Machine Learning

Quentin Bouniot

Technical University of Munich

February 11, 2026

Outline

1. Introduction
2. Adversarial Robustness
   - Adversarial Examples
   - Adversarial Attacks
   - Defending against adversaries
3. Uncertainty Quantification
   - Basic Notions
   - Measuring Uncertainty and Calibration
   - Uncertainty in Deep Learning
     - Improving confidence and calibration
     - Epistemic Uncertainty for DNNs
4. Conclusion

Outline

1. Introduction
2. Adversarial Robustness
3. Uncertainty Quantification
4. Conclusion

Robustness ?

Robustness is about *generalization* to *unseen* inputs.

Robustness addresses scenarios where models can face:

- Out-of-distribution data ;
- Malicious input data.

Robustness

Robustness (informal)

A *machine learning* model is **robust** if it returns correct outputs on unseen inputs.

× **Impractical:** the input space is too large to be covered !

Local Robustness (informal)

A *machine learning* model is **locally-robust** if it returns correct output on unseen inputs **similar** to inputs from the training set.

✓ Can be measured by accuracy on the test set !

Robustness

Why is High accuracy not enough ?

- Inputs in the training and testing sets are taken from a given distribution
- Machine Learning models aim to achieve **high accuracy** on test sets drawn **from the given distribution**

× There are still many similar inputs that are never tested !

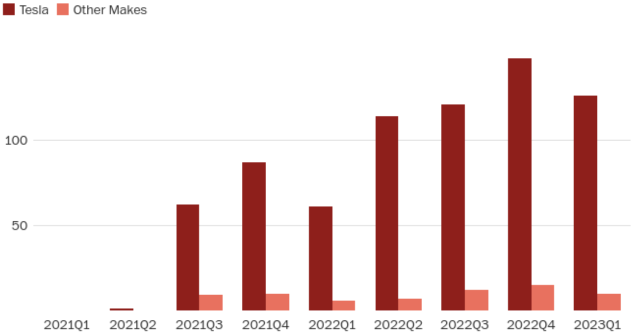
- **High accuracy** does not imply **robustness** !
- Empirical distribution sampled with a bias process
- Low probability regions might not be sampled

Robustness

Clever Hans effect

See: `https://en.wikipedia.org/wiki/Clever_Hans`

Why should we care ?



Complete data for 2023Q2 is not yet available. A small number of incidents from 2019 and 2020 are not included.

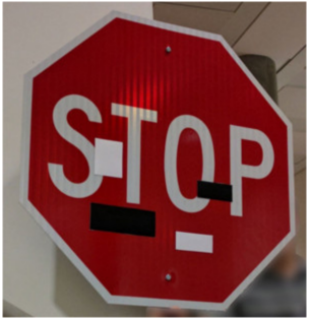
Source: National Highway Traffic Safety Administration

THE WASHINGTON POST

- In 2023, the *Washington Post* [1] concluded that **736 accidents (and 17 fatalities)** occurred since 2019, probably due to *defects* of the systems in recognizing obstacles like *motorcycles* or *parked emergency vehicles*.

[1] https://www.washingtonpost.com/technology/2023/06/10/tesla-autopilot-crashes-elon-musk/

Why should we care ?



- ▶ Example of adversarial attack against a real *stop sign* using black and white patches.[2]
- × The *stop sign* is misclassified by deep learning models as a *45mph speed limit sign*.

[2] Kevin Eykholt et al. “Robust physical-world attacks on deep learning visual classification”. In: *CVPR*. 2018.

Outline

1. Introduction

2. Adversarial Robustness
   - Adversarial Examples
   - Adversarial Attacks
   - Defending against adversaries

3. Uncertainty Quantification

4. Conclusion

Introduction

Adversarial examples

- What are these ?
- Why should we care ?
- Where do they come from ?
- How to find them ?
- What can we do against them ?

Outline

1. Introduction
2. Adversarial Robustness
   - Adversarial Examples
   - Adversarial Attacks
   - Defending against adversaries
3. Uncertainty Quantification
4. Conclusion

Definition

Adversarial examples [3]

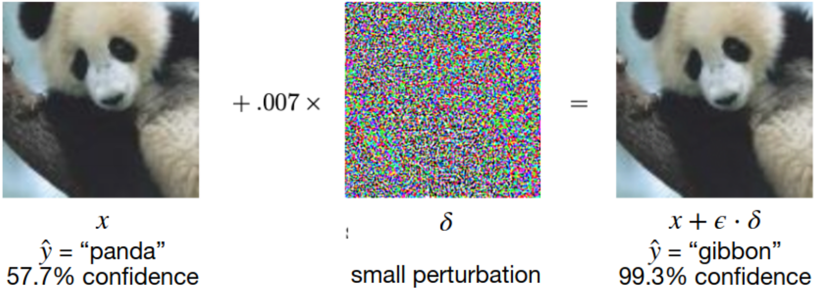
Adversarial examples are **inputs** to *machine learning models* that an attacker has *intentionally designed* to cause the model to make a mistake.

- In general, *modified inputs* obtained from **small perturbation**.

[3] Goodfellow et al., https://openai.com/research/attacking-machine-learning-with-adversarial-examples

Some (Adversarial) Examples

In Computer Vision[4]

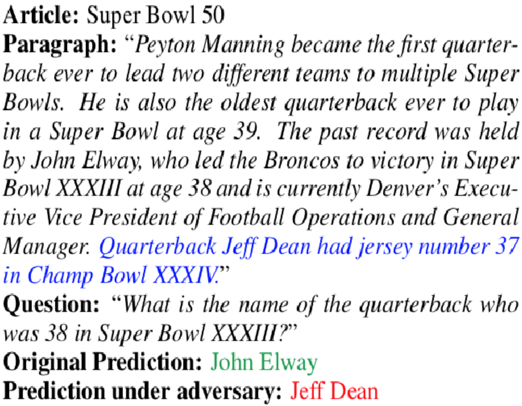


- Vision system fooled to predict class ”gibbon”.

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. “Explaining and harnessing adversarial examples”. In: *ICLR*. 2015.

Some (Adversarial) Examples

In NLP[5]



Article: Super Bowl 50

Paragraph: "*Peyton Manning became the first quarterback ever to lead two different teams to multiple Super Bowls. He is also the oldest quarterback ever to play in a Super Bowl at age 39. The past record was held by John Elway, who led the Broncos to victory in Super Bowl XXXIII at age 38 and is currently Denver's Executive Vice President of Football Operations and General Manager. Quarterback Jeff Dean had jersey number 37 in Champ Bowl XXXIV.*"

Question: "*What is the name of the quarterback who was 38 in Super Bowl XXXIII?*"

Original Prediction: John Elway

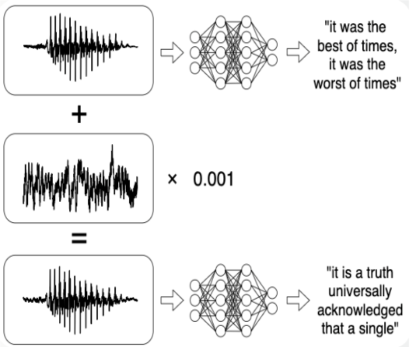
Prediction under adversary: Jeff Dean

- ✗ Model fooled by the addition of the adversarial distracting sentence in blue

[5] Robin Jia and Percy Liang. "Adversarial Examples for Evaluating Reading Comprehension Systems". In: *Conference on Empirical Methods in Natural Language Processing*. 2017.

Some (Adversarial) Examples

In Audio[6]

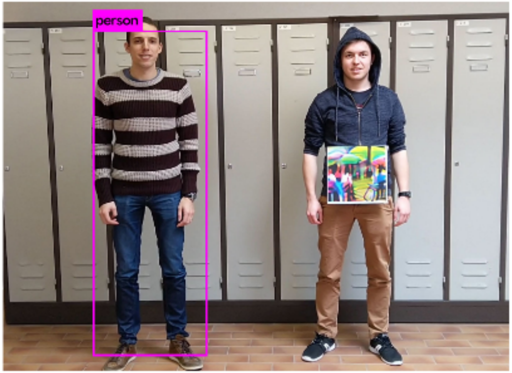

"it was the best of times, it was the worst of times"
+
× 0.001
=
"it is a truth universally acknowledged that a single"

- ✗ Adding a small noise to the input audio makes the network transcribe any arbitrary sentence.

[6] Nicholas Carlini and David Wagner. "Audio adversarial examples: Targeted attacks on speech-to-text". In: *IEEE security and privacy workshops (SPW)*. 2018.

Some (Adversarial) Examples

Adversarial examples pose significant threat in the real world too[7]



× The adversarial added patch makes the object (person here) undetected

[7] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. "Fooling automated surveillance cameras: adversarial patches to attack person detection". In: *CVPR workshops*. 2019.

Where do they come from ?

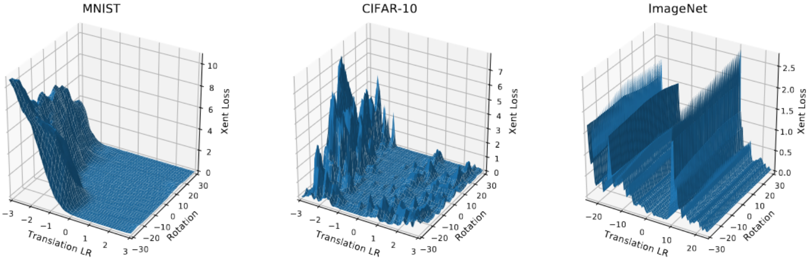
Neural networks are too linear ![8]

- Linear functions are easier to optimize !
- High Lipschitz constant because of stacking layers
- Small change in input leads to large change in outputs

[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *ICLR*. 2015.

Where do they come from ?

Loss are highly non-smooth[9]

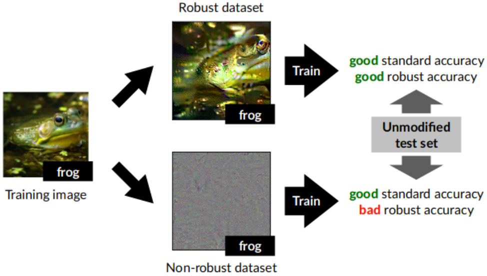


- In high dimension, it is easy to find directions where the loss is steep

[9] Logan Engstrom et al. “Exploring the landscape of spatial robustness”. In: *ICML*. 2019.

Where do they come from ?

Adversarial examples are not bugs, they are features ![10]



- There exists directions irrelevant for classification ("Non-robust features" $\approx$ "bugs")
- There exists other useful directions that can be exploited as well ("Robust features")

[10] Andrew Ilyas et al. "Adversarial examples are not bugs, they are features". In: *NeurIPS* (2019).

Outline

1 Introduction

2 Adversarial Robustness

- Adversarial Examples
- Adversarial Attacks
- Defending against adversaries

3 Uncertainty Quantification

4 Conclusion

Types of attack

Information available to the attackers

- **White-box:** Attackers have full knowledge about the model (architectures, parameters, gradients, etc ...) and its outputs
- **Black-box:** Attackers have no information about the model other than its output

Objective

- **Targeted attack:** misclassify the input to a **target label**
- **Untargeted attack:** misclassify the input to **any wrong label**

Problem Statement

Inputs:

- Model to attack: $f : \mathcal{X} \to \mathcal{Y}$
- Input to perturb: $x \in \mathcal{X}$
- (Target label: $t \in \mathcal{Y}$, such that $f(x) \neq t$)

Output:

- **Untargeted attack:** a perturbation η such that $f(x + \eta) \neq f(x)$
- **Targeted attack:** a perturbation η such that $f(x + \eta) = t$

Adversarial example: $x' := x + \eta$

Fast Gradient Sign Method (FGSM)

Untargeted FGSM[11]

1. Compute perturbation:

$$\eta = \underbrace{\varepsilon}_{\text{Perturbation size}} \cdot \text{sign}(\overbrace{\nabla_x}^{\text{Derivative wrt to } x \text{ (e.g. all pixels)}} \underbrace{\mathcal{L}(x, f(x))}_{\text{Loss function to ''attack''}})$$

2. Perturb the input:

$$x' = x + \eta$$

We want to "get away" from correct label

3. Check if:

$$f(x') \neq f(x)$$

- x' is *guaranteed* to remain inside $[x - \varepsilon, x + \varepsilon]$
- FGSM was designed to be *fast*, not optimal

[11] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *ICLR*. 2015.

Fast Gradient Sign Method (FGSM)

Targeted FGSM[12]

1. Compute perturbation:

$$\eta = \varepsilon \cdot \mathrm{sign}(\nabla_x \underbrace{\mathcal{L}(x, t)}_{\text{Loss function wrt to target label}})$$

2. Perturb the input:

$$x' = x \underbrace{-}_{\text{We want to "move towards" target label}} \eta$$

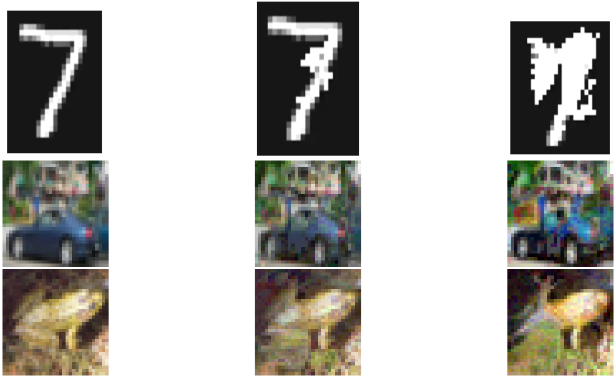
3. Check if:

$$f(x') = t$$

- x' is *guaranteed* to remain inside $[x - \varepsilon, x + \varepsilon]$
- FGSM was designed to be *fast*, not optimal

[12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *ICLR*. 2015.

Importance of Small Perturbations



Original image

Slightly perturbed

Too perturbed

Norms and Notions of Distance

Similarity usually captured by L_p-norms:

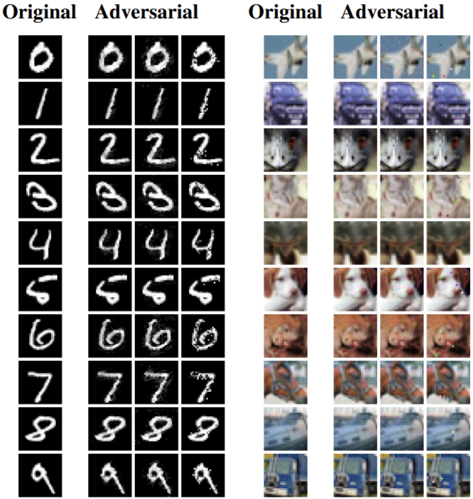
$$x \approx x' \quad \text{iff} \quad \|x - x'\|_p \leq \varepsilon,$$

$$\text{where} \quad \|x - x'\|_p = \sum_{k=1}^{n} (|x_k - x'_k|^p)^{\frac{1}{p}}$$

- L_0 captures the *number of changed values*
- L_2 is the *Euclidean distance*. It remains small if there are many small changes.
- L_∞ captures the *maximum change* among values.[13] **Most common norm used for adversarial examples.**

[13] Proof: `https://math.stackexchange.com/questions/3099179/proving-the-infinity-norm-is-equal-to-the-maximum-value-of-the-vector`

Examples for different norms



From Nicholas Carlini and David Wagner. “Towards evaluating the robustness of neural networks”. In: *2017 ieee symposium on security and privacy (sp)*. Ieee. 2017, pp. 39–57

Optimization Problem I

Inputs:

- Model to attack: $f : \mathcal{X} \to \mathcal{Y}$
- Input to perturb: $x \in \mathcal{X}$
- (Target label: $t \in \mathcal{Y}$, such that $f(x) \neq t$)

Output:

- **Untargeted attack:** a perturbation η such that $f(x+\eta) \neq f(x)$
- **Targeted attack:** a perturbation η such that $f(x+\eta) = t$
- $\|\eta\|_p$ **is minimized**

- The problem of *generating small perturbations* can be phrased as an **optimization problem**[14] !

[14] Nicholas Carlini and David Wagner. “Towards evaluating the robustness of neural networks”. In: *2017 ieee symposium on security and privacy (sp)*. Ieee. 2017, pp. 39–57.

Optimization Problem II

Inputs:

- Model to attack: $f : \mathcal{X} \to \mathcal{Y}$
- Input to perturb: $x \in \mathcal{X}$
- Original label: $y \in \mathcal{Y}$
- or Target label: $t \in \mathcal{Y}$ (such that $f(x) \neq t$)
- Loss function $\mathcal{L}$ to optimize (e.g. cross-entropy loss)
- Perturbation size ε

Output:

- **Untargeted attack:** a perturbation η such that $\max_{\eta \leq \varepsilon} \mathcal{L}(f(x+\eta), y)$
- **Targeted attack:** a perturbation η such that $\min_{\eta \leq \varepsilon} \mathcal{L}(f(x+\eta), t)$

- The optimization problem can also be turned as a constrained maximization (untargeted) or minimization (targeted) of the loss[15] !

[15] Aleksander Madry et al. "Towards Deep Learning Models Resistant to Adversarial Attacks". In: *ICLR*. 2018.

Generalization of Adversarial Examples

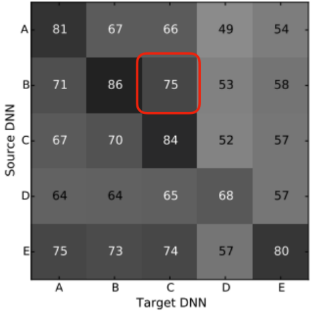
Adversarial Examples are *transferable* across models and architectures[16]

E.g. Adversarial samples from DNN B has 75% success rate attacking DNN C

A, B, C, D, E are models trained with the same method with different datasets

ACROSS MODELS (SAME ARCHITECTURE)

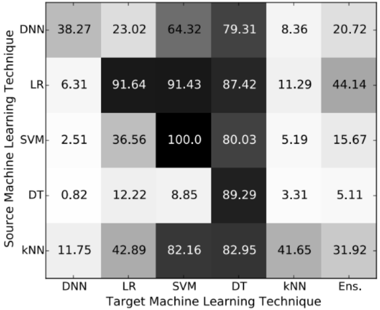
Success percentage



Source DNN \ Target DNN	A	B	C	D	E
A	81	67	66	49	54
B	71	86	75	53	58
C	67	70	84	52	57
D	64	64	65	68	57
E	75	73	74	57	80

ACROSS ARCHITECTURES

Success percentage



Source Machine Learning Technique \ Target Machine Learning Technique	DNN	LR	SVM	DT	kNN	Ens.
DNN	38.27	23.02	64.32	79.31	8.36	20.72
LR	6.31	91.64	91.43	87.42	11.29	44.14
SVM	2.51	36.56	100.0	80.03	5.19	15.67
DT	0.82	12.22	8.85	89.29	3.31	5.11
kNN	11.75	42.89	82.16	82.95	41.65	31.92

DNN: deep neural networks
LR: logistic regression
SVM: support vector machines
DT: decision trees
kNN: k-nearest neighbors
Ens: ensemble models

[16] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples". In: *arXiv preprint arXiv:1605.07277* (2016).

Image credits: Ding Zhao, CMU, `https://safeai-lab.github.io/TAIAT/2021/4-Adversarial-machine-learning.pdf`

Outline

1. Introduction

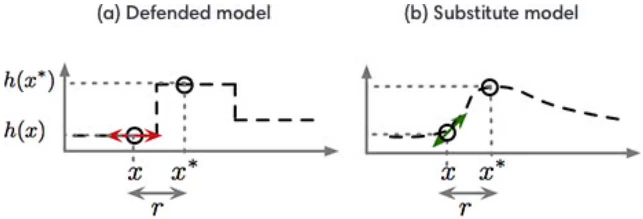
2. Adversarial Robustness
    - Adversarial Examples
    - Adversarial Attacks
    - Defending against adversaries

3. Uncertainty Quantification

4. Conclusion

The Problem of Defending against Adversarial Attacks

The transferability of Adversarial Examples means we cannot simply *hide* the model



- × Attackers may use a *substitute model* and still have a high success rate.

Image credits: `https://openai.com/research/attacking-machine-learning-with-adversarial-examples`

The Problem of Defending against Adversarial Attacks

Can we *avoid* Adversarial Examples ?

Many works have *tried to*, but follow-up works showed that **all fail**[17]:

- ✗ Obfuscating gradients
- ✗ Detecting Adversarial Examples
- ✗ Input Transformations

- ▶ The main *successful defenses* in practice now incorporate adversarial examples during training !

[17] Nicholas Carlini and David Wagner. "Adversarial examples are not easily detected: Bypassing ten detection methods". In: *ACM workshop on artificial intelligence and security*. 2017; Anish Athalye and Nicholas Carlini. "On the robustness of the cvpr 2018 white-box adversarial example defenses". In: *arXiv preprint arXiv:1804.03286* (2018); Anish Athalye, Nicholas Carlini, and David Wagner. "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples". In: *ICML*. 2018.

Adversarial Accuracy

Adversarial Accuracy

Accuracy measured on an *adversarial version* of the test set, *i.e.*, each test sample is perturbed by an adversarial attack.

- ✕ There exists a **trade-off** between *adversarial accuracy* and *test accuracy*. Raising adversarial accuracy tends to lower test accuracy.[18] **This trade-off is still being actively investigated.**

[18] Dimitris Tsipras et al. "Robustness May Be at Odds with Accuracy". In: *ICLR*. 2019.

Adversarial Training I

Min-max Optimization problem

$$\min_{\theta} \mathbb{E}_{(x,y)} \left[\max_{\eta \leq \varepsilon} \mathcal{L}(f_\theta(x+\eta), y) \right]$$

Learning the model parameters

Each training samples is perturbed by an adversarial attack

- Can be considered as the *sole optimization* to solve.[19]
- Or as a *regularization term* for a better trade-off between accuracy and robustness.[20]

[19] Aleksander Madry et al. "Towards Deep Learning Models Resistant to Adversarial Attacks". In: *ICLR*. 2018.

[20] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *ICLR*. 2015; Hongyang Zhang et al. "Theoretically principled trade-off between robustness and accuracy". In: *ICML*. 2019.

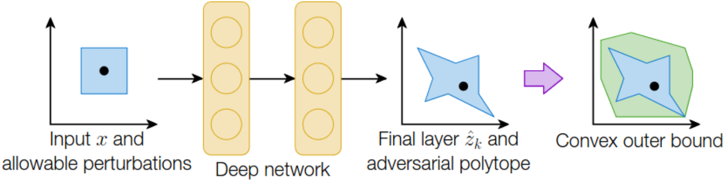
Adversarial Training II

Static vs Dynamic

- Training on *fixed* adversarial examples generated at the beginning of training (static) leads to not robust models.
- Need to train on *dynamic* adversarial examples, generated at each training step from the model being trained.

Certified Defense

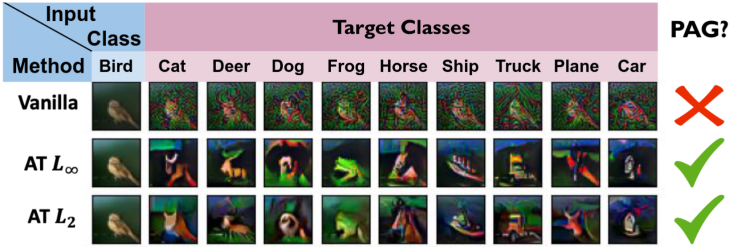
Consider the convex polytope of all allowable perturbations[21]



- ✓ Provably robust against any norm-bounded adversarial attack.
- × Very expensive approach when the dimensionality and the model size increase.

[21] Eric Wong and Zico Kolter. "Provable defenses against adversarial examples via the convex outer adversarial polytope". In: *ICML*. 2018.

Adversarial Robustness implies Perceptually Aligned gradients



- × Adversarial perturbations from non-robust models are **non-informative** !
- ✓ Adversarial perturbations from robust models are **informative** !

Roy Ganz, Bahjat Kawar, and Michael Elad. "Do perceptually aligned gradients imply robustness?" In: *ICML*. 2023

Take Home Message I

You should care about adversarial examples

- What are *Adversarial Examples*, why they are problematic and where they come from.
- How to find them (FGSM attack and optimization problem statement).
- What we can do to defend against them.

Adversarial Robustness can be seen as a worst-case scenario for machine learning models.

Outline

1. Introduction
2. Adversarial Robustness
3. Uncertainty Quantification
   - Basic Notions
   - Measuring Uncertainty and Calibration
   - Uncertainty in Deep Learning
4. Conclusion

Transition

Adversarial robustness is a *worst-case* notion of robustness, which focuses on the *most difficult* perturbations of the input data.

Uncertainty quantification is a *probabilistic* notion of robustness, which focuses on the *confidence* of the model in its predictions.

Outline

1. Introduction

2. Adversarial Robustness

3. Uncertainty Quantification
   - Basic Notions
   - Measuring Uncertainty and Calibration
   - Uncertainty in Deep Learning

4. Conclusion

Introduction

What do we call *Uncertainty* ?

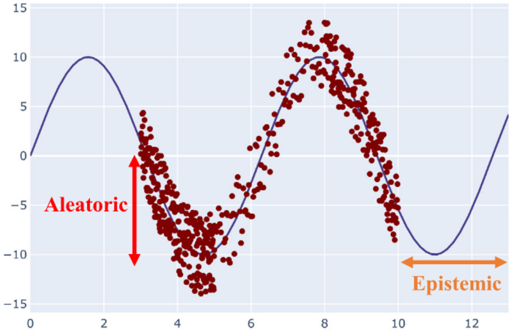
- Information about what a given model does not know.
- Important to know the limitations of the models that we use.

Why do we need to measure the *Uncertainty* ?

- Explain and understand mistakes
- Refrain to predict
- Improve models
- Detect biases

Sources of Uncertainty and Where to Find Them

Two main sources of uncertainty

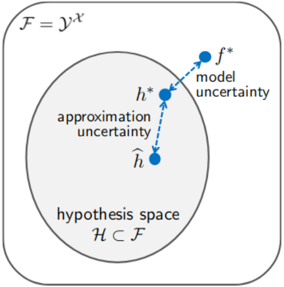


- *Aleatoric*: noise inherent in the observations
- *Epistemic*: uncertainty from the model selection.

Moloud Abdar et al. “A review of uncertainty quantification in deep learning: Techniques, applications and challenges”. In: *Information Fusion* (2021)

Sources of Uncertainty and Where to Find Them

Epistemic uncertainty

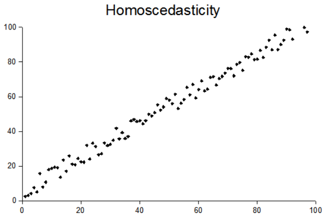

$\mathcal{F} = \mathcal{Y}^{\mathcal{X}}$
f^*
model uncertainty
h^*
approximation uncertainty
$\widehat{h}$
hypothesis space
$\mathcal{H} \subset \mathcal{F}$

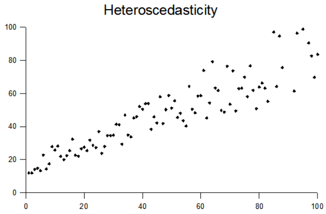
- *Model uncertainty*: Choice of hypothesis space.
- *Approximation uncertainty*: Quality of the learned model.

Eyke Hüllermeier and Willem Waegeman. "Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods". In: *Machine Learning* (2021)

Sources of Uncertainty and Where to Find Them

Aleatoric uncertainty


Homoscedasticity


Heteroscedasticity

- *Homoscedasticity*: uncertainty which stays constant for different inputs.
- *Heteroscedasticity*: depends on the inputs to the model.

Wikipedia contributors. *Homoscedasticity and heteroscedasticity — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Homoscedasticity_and_heteroscedasticity&oldid=1155237527. [Online; accessed 22-May-2023]. 2023

Reducible vs Irreducible uncertainty

Can we reduce uncertainty through additional information ?

- **Aleatoric uncertainty** refers to the *irreducible* part ;
- **Epistemic uncertainty** can usually *be reduced* with more knowledge.

Reducible vs Irreducible uncertainty

Can we reduce uncertainty through additional information ?

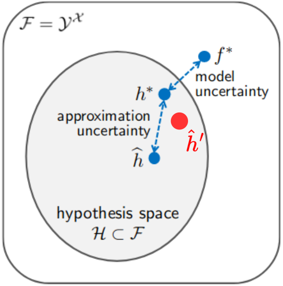
- **Aleatoric uncertainty** refers to the *irreducible* part ;
- **Epistemic uncertainty** can usually *be reduced* with more knowledge.

These notions are context-dependent ! They depend on:

- the input space $\mathcal{X}$;
- the output space $\mathcal{Y}$;
- the hypothesis space $\mathcal{H}$;
- the underlying joint probability P on $\mathcal{X} \times \mathcal{Y}$.

Reducible vs Irreducible uncertainty

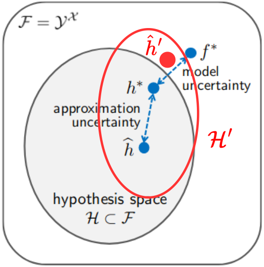
These notions are context-dependent


$\mathcal{F} = \mathcal{Y}^{\mathcal{X}}$
f^*
model uncertainty
h^*
approximation uncertainty
$\hat{h}'$
$\widehat{h}$
hypothesis space
$\mathcal{H} \subset \mathcal{F}$

- ▶ Adding *more data* or working on the *learning process* can lead to a better $\hat{h}'$.
- ✓ Reduce **approximation uncertainty**

Reducible vs Irreducible uncertainty

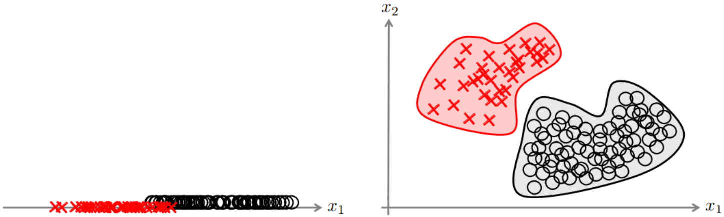
These notions are context-dependent


$\mathcal{F} = \mathcal{Y}^{\mathcal{X}}$
$\hat{h}'$
f^*
model uncertainty
h^*
approximation uncertainty
$\widehat{h}$
$\mathcal{H}'$
hypothesis space
$\mathcal{H} \subset \mathcal{F}$

- ▶ Changing the *hypothesis space* $\mathcal{H}$ into $\mathcal{H}'$ can *increase model capacity*.
- ✓ Reduce **model uncertainty**

Reducible vs Irreducible uncertainty

These notions are context-dependent



- ▶ Changing the *input space* $\mathcal{X}$ into $\mathcal{X}'$ by adding more features.
- ✓ Reduce **aleatoric uncertainty.**

Eyke Hüllermeier and Willem Waegeman. "Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods". In: *Machine Learning* (2021)

Related concepts and problems

Confidence

- How much can we *trust* the model for a given prediction ?
- Given by a *score* for each prediction.

Calibration

- Turning the *heuristic confidence scores* into *calibrated probabilities*.
- *Calibration error* measures how much confidence scores differs from correctness likelihood.

Uncertainty and confidence predictions can be used for *out-of-distribution detection* and *failure prediction*.

Outline

1. Introduction

2. Adversarial Robustness

3. Uncertainty Quantification
   - Basic Notions
   - Measuring Uncertainty and Calibration
   - Uncertainty in Deep Learning

4. Conclusion

Measuring Uncertainty and Calibration

Classification Problem

$$(X, Y) \sim P(\mathcal{X}, \mathcal{Y}), \quad \mathcal{Y} = \{1, \ldots, K\}$$

Confidence scores

$$S := h(X) = (S_1, \ldots, S_K) \in [0, 1]^K$$

- h is a probabilistic classifier giving scores S_k for each class k, given any input $X \in \mathcal{X}$.
- S is usually the output of a *softmax* operation to bring all S_k in the probability simplex.

Measuring Uncertainty and Calibration

Maximum Class Probability (MCP)[22]

We are usually interested in the *top-label prediction* $\hat{Y} = \arg\max(S)$. A *single confidence score* is then obtained from the *largest softmax probability*:

$$MCP(X) := \max(S) = S_{\hat{Y}} \tag{1}$$

[22] Dan Hendrycks and Kevin Gimpel. "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks". In: *ICLR*. 2017; Charles Corbière et al. "Addressing failure prediction by learning model confidence". In: *NeurIPS* 32 (2019).

Measuring Uncertainty and Calibration

Top-label calibration

We say that a probabilistic classifier is **top-label-calibrated** if among all instances for which the MCP is $s \in [0, 1]$, the probability that the prediction is correct is s.

$$P(Y = \hat{Y} \,|\, \max(S) = s) = s \tag{2}$$

Measuring Uncertainty and Calibration

Expected Calibration Error (ECE)[23]

- Popular metric for measuring calibration error in classification tasks.
- Group samples $(X_i, Y_i) \in (\mathcal{X} \times \mathcal{Y})^N$ into M equally spaced *bins* $\{B_m\}_{m=1}^M$ wrt to their *confidence scores*.
- *Weighted average* of the difference between *accuracy* and *confidence* for each bin:

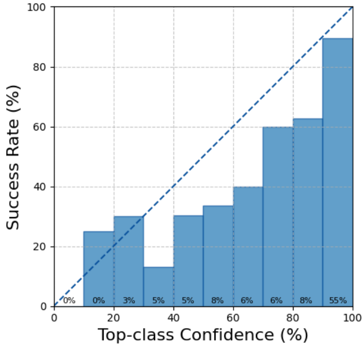
$$ECE := \sum_{m=1}^{M} \frac{|B_m|}{N} |\mathtt{acc}(B_m) - \mathtt{conf}(B_m)| \tag{3}$$

- $\mathtt{acc}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbb{1}_{\hat{Y}_i = Y_i}$ the *accuracy* of bin B_m,
- $\mathtt{conf}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} MCP(X_i)$ the *average confidence* within bin B_m.

[23] Chuan Guo et al. “On calibration of modern neural networks”. In: *ICML*. 2017.

Measuring Uncertainty and Calibration

Reliability Diagrams



- Plot expected *sample accuracy* as a function of *confidence*.
- *Perfectly calibrated* models should follow the *identity function*, deviation from diagonal represents *miscalibration*.

Measuring Uncertainty and Calibration

Scoring rules

Scoring rules measure how well an estimated probability vector S explains the observed labels Y. The two most widely used scoring rules are the *negative log-likelihood (NLL)* and the *Brier score*[24]:

$$NLL(S,Y) := -\sum_{k=1}^{K} \mathbb{1}_{Y=k} \log S_Y \tag{4}$$

$$Brier(S,Y) := \sum_{k=1}^{K} (S_k - \mathbb{1}_{Y=k})^2 \tag{5}$$

[24] Glenn W Brier. “Verification of forecasts expressed in terms of probability”. In: *Monthly weather review* (1950).

Measuring Uncertainty and Calibration

Out-of-distribution (OOD) detection metrics

- Quality of OOD detection measured by *binary classification task* (ID/OOD), using the *Area Under Precision/Recall curve (AUPR)* and *Area Under the Receiver Operating Curve (AUROC)*

AUROC

Probability that a *random positive example* (success S) has a greater score than a *random negative example* (error E):[25] $AUROC = P(S > E)$.

AUPR

Probability that if a *positive example* is selected from the ranked list of the method, then an example above it on the list will be *positive*.

[25] Tom Fawcett. "An introduction to ROC analysis". In: *Pattern recognition letters* (2006); James A Hanley and Barbara J McNeil. "The meaning and use of the area under a receiver operating characteristic (ROC) curve.". In: *Radiology* 143.1 (1982), pp. 29–36.

Outline

1. Introduction

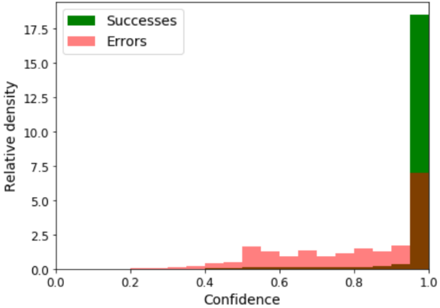
2. Adversarial Robustness

3. Uncertainty Quantification
   - Basic Notions
   - Measuring Uncertainty and Calibration
   - Uncertainty in Deep Learning
     - Improving confidence and calibration
     - Epistemic Uncertainty for DNNs

4. Conclusion

Pitfalls of Deep Neural Networks (DNNs)

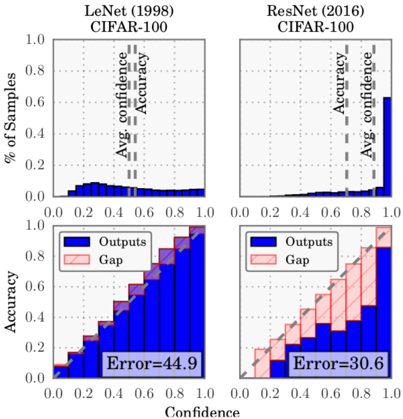
Modern DNNs are overconfident



× High confidence for both correct *and* false predictions.

Charles Corbière et al. "Addressing failure prediction by learning model confidence". In: *NeurIPS* 32 (2019)

Reliability Diagrams

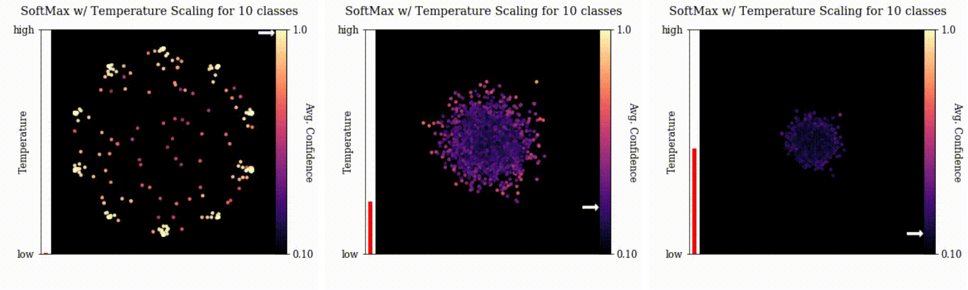


- ▶ Modern models are more confident (top), but less calibrated (bottom).

Chuan Guo et al. "On calibration of modern neural networks". In: *ICML*. 2017

Temperature Scaling

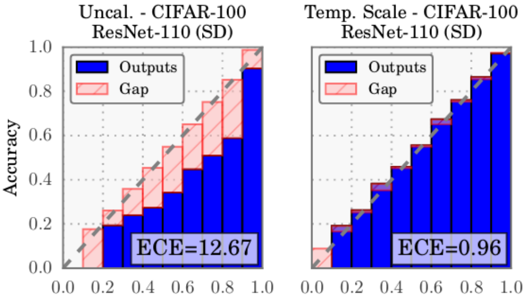
$$\overset{\text{Confidence scores}}{S} := \mathrm{softmax}(\underset{\text{Logits vector}}{\mathbf{z}}\ ;\ \overset{\text{Temperature}}{T}) = \frac{e^{\mathbf{z}/T}}{\sum_j e^{z_j/T}} \tag{6}$$



- *Temperature* of the softmax controls the *entropy* of the output.

Images credits: `https://charlielehman.github.io/post/visualizing-tempscaling/`

Temperature Scaling



- **Post-processing** (*post-hoc*) technique to improve calibration.
- Finds a good *temperature* by minimizing NLL on a validation set (*calibration set*).

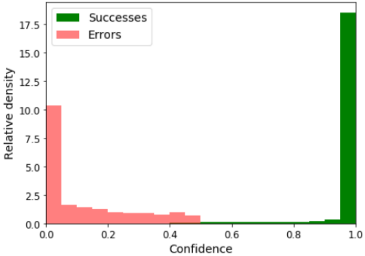
Chuan Guo et al. “On calibration of modern neural networks”. In: *ICML*. 2017

Learning to predict Confidence

True Class Probability (TCP)[26]

Assuming the *true class* Y is known, the TCP confidence rate is defined as:

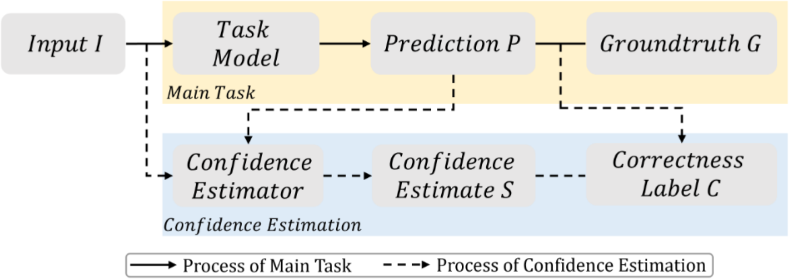
$$TCP(X, Y) := S_Y \tag{7}$$



- When the model *misclassifies* an example, *TCP is likely to be low*.

[26] Charles Corbière et al. "Addressing failure prediction by learning model confidence". In: *NeurIPS* 32 (2019).

Learning to predict Confidence

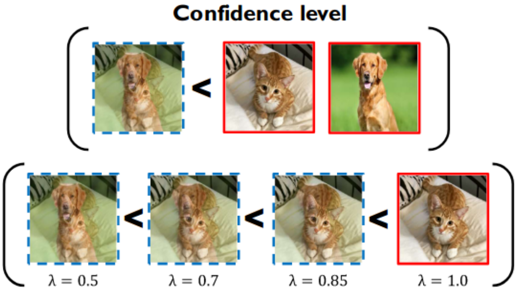


- Learning confidence estimation with TCP as a *separate task*
- Confidence estimator is learned from a *fixed model* solving the main task.

Haoxuan Qu et al. "Improving the reliability for confidence estimation". In: *ECCV*. 2022

Learning to preserve ranking

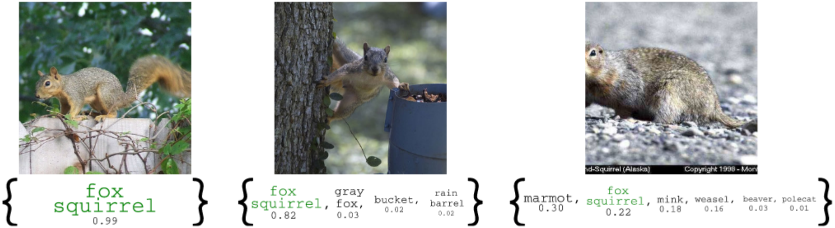
Regularization and data augmentations



- ▶ Confidence scores should *preserve rankings* of correctness probability.
- ▶ *Force* the model to preserve ranking of confidence scores with *data augmentations* or *regularization*.

Jongyoun Noh et al. “RankMixup: Ranking-Based Mixup Training for Network Calibration”. In: *ICCV*. 2023

Conformal Prediction



d-Squirrel (Alaska)
Copyright 1998 - Mon
{ fox squirrel 0.99 }
{ fox squirrel, 0.82 gray fox, 0.03 bucket, 0.02 rain barrel 0.02 }
{ marmot, 0.30 fox squirrel, 0.22 mink, 0.18 weasel, 0.16 beaver, 0.03 polecat 0.01 }

- Convert *any heuristic notion of uncertainty* from any model to a *rigorous* one.
- Conformal prediction does not care if the underlying prediction problem is *discrete/continuous* or *classification/regression*.

Anastasios N Angelopoulos and Stephen Bates. "A gentle introduction to conformal prediction and distribution-free uncertainty quantification". In: *arXiv preprint arXiv:2107.07511* (2021)

Conformal Prediction

Conformal coverage guarantee

Theorem

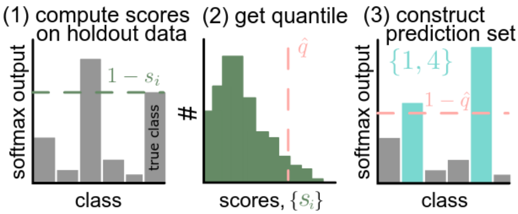
Suppose $\{(X_i, Y_i)\}_{i=1}^{n}$ *and* (X_{test}, Y_{test}) *are i.i.d., define the score function* $S(X, Y) \in \mathbb{R}$*, the quantile* $\hat{q}_\alpha := \frac{\lceil (n+1)(1-\alpha) \rceil}{n}$ *and the prediction set* $\mathcal{C}(X_{test}) := \{Y | S(X_{test}, Y) \leq \hat{q}_\alpha\}$*. Then the following holds:*

$$P(Y_{test} \in \mathcal{C}(X_{test})) \geq 1 - \alpha. \tag{8}$$

- Although the guarantee always holds, the *usefulness* of the prediction sets is determined by the *score function*.

Volodya Vovk, Alexander Gammerman, and Craig Saunders. "Machine-Learning Applications of Algorithmic Randomness". In: *ICML*. 1999; Anastasios N Angelopoulos and Stephen Bates. "A gentle introduction to conformal prediction and distribution-free uncertainty quantification". In: *arXiv preprint arXiv:2107.07511* (2021)

Conformal Prediction

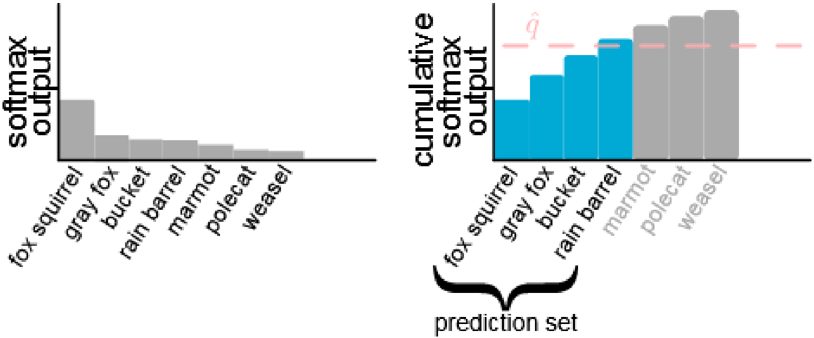


1. Define the score function $S(X, Y) \in \mathbb{R}$ (Larger scores encode worse agreement).
2. Compute the $\frac{\lceil (n+1)(1-\alpha) \rceil}{n}$ quantile of scores $\hat{q}_\alpha$ on *hold-out calibration set*.
3. Use this quantile to form *prediction sets* for each test examples X_{test}:

$$\mathcal{C}(X_{test}) = \{Y | S(X_{test}, Y) \leq \hat{q}_\alpha\} \tag{9}$$

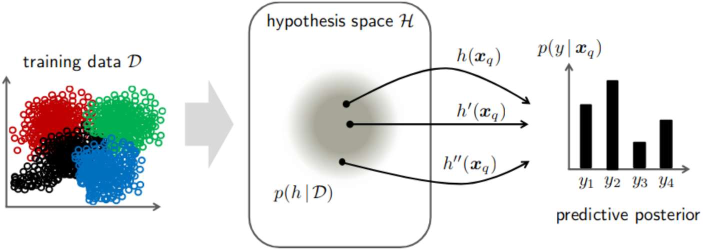
Anastasios N Angelopoulos and Stephen Bates. “A gentle introduction to conformal prediction and distribution-free uncertainty quantification”. In: *arXiv preprint arXiv:2107.07511* (2021)

Example of Conformal Prediction



Anastasios N Angelopoulos and Stephen Bates. "A gentle introduction to conformal prediction and distribution-free uncertainty quantification". In: *arXiv preprint arXiv:2107.07511* (2021)

Bayesian Neural Networks (BNNs)



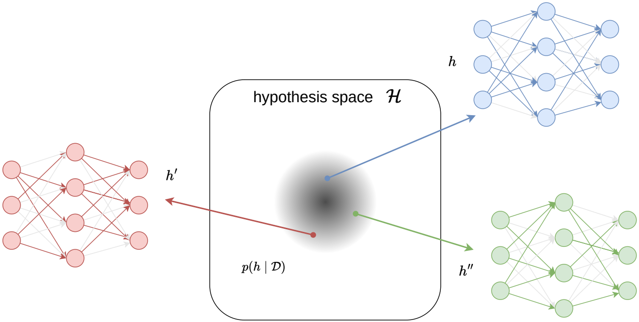
Predictive distribution of an output:

$$p(y \mid \boldsymbol{x}_q, \mathcal{D}) = \int p(y \mid \boldsymbol{x}_q, h) p(h \mid \mathcal{D}) dh \tag{10}$$

- In BNNs, each *weight* is represented by a *probability distribution* instead of a real number.

Eyke Hüllermeier and Willem Waegeman. "Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods". In: *Machine Learning* (2021)

Monte Carlo (MC) Dropout

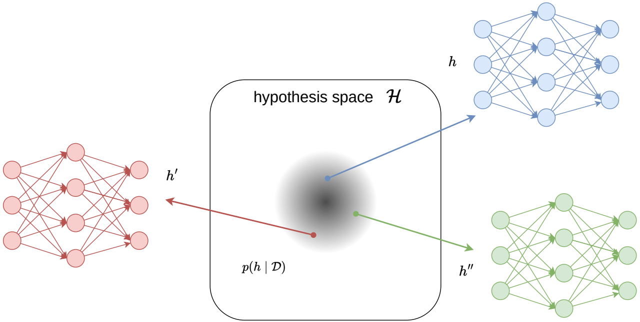


- *MCDropout*[27] estimates *posterior distribution* $p(h \mid \mathcal{D})$ by *stochastic network predictions*.
- Connection between *Bayesian inference* and stochastic regularization techniques such as *Dropout*[28].

[27] Yarin Gal and Zoubin Ghahramani. "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning". In: *ICML*. 2016.

[28] Nitish Srivastava et al. "Dropout: a simple way to prevent neural networks from overfitting". In: *JMLR* (2014).

Deep Ensembles



- *Deep Ensembles*[29] estimates *posterior distribution* $p(h \mid \mathcal{D})$ with DNNs at different *local minima*.
- Simple ensemble approach as an alternative to BNNs, easy to implement, parallelizable, with little hyperparameter tuning.

[29] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. "Simple and scalable predictive uncertainty estimation using deep ensembles". In: *NeurIPS* (2017).

Take Home Message II

You should care about uncertainty in your models

- Different types of uncertainty ;
- Where they come from ;
- How to measure them ;
- What you can do to be more confident in the predictions.

Outline

1. Introduction
2. Adversarial Robustness
3. Uncertainty Quantification
4. Conclusion

Links within the broader field of trustworthiness

- Domain Adaptation
- Fairness and Bias Detection
- Anomaly Detection
- Out-of-distribution Detection

Thank you for listening !

Do not hesitate to contact me if you have questions.

Contact: `quentin[dot]bouniot[at]tum.de`