Adversarial Uncertainty Learning in Deep Neural Networks

Łukasz Grad

University of Warsaw

l.grad@uw.edu.pl

October 29, 2021

Overview

Kinds of Uncertainty

- **Aleatoric (Data) Uncertainty**
  - Inherent to the problem
  - Caused by information loss when representing the real world within a data sample
  - Irreducible
- **Epistemic (Model) Uncertainty**
  - Uncertainty in the estimated model parameters
  - Caused by lack of data, errors (noise) in training procedure, insufficient model structure
  - Reducible
- **Total (Prediction) Uncertainty**
  - Combines epistemic and aleatoric uncertainty

Uncertainty - Regression

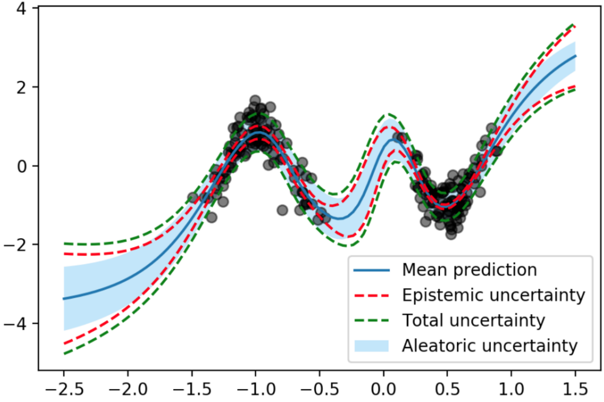


Figure: Clements, William R., et al. "Estimating risk and uncertainty in deep reinforcement learning."

Uncertainty - Classification

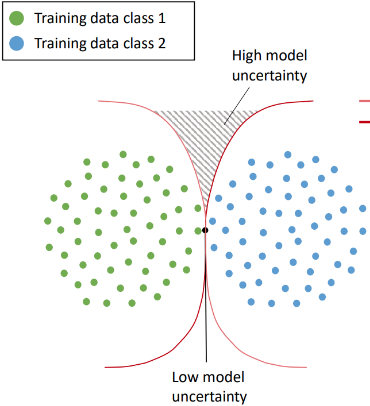


Figure: Gawlikowski, Jakob, et al. "A survey of uncertainty in deep neural networks."

Uncertainty - use cases

- Misclassification detection
  - Uncertain predictions are less likely to be correct
- Out of distribution detection
  - Instances from a different distribution should have high epistemic uncertainty
- Adversarial input detection
  - Should adversarial inputs have high uncertainty?

Uncertainty Measures - Classification

Given a dataset D and a function $f(x,\theta)$ we model the outcome

$$p(y|x,\theta) = Cat(y; f(x,\theta))$$

During inference we also obtain $p(\theta|D)$. We estimate predictive distribution

$$p(y|x,D) = \mathbb{E}_{p(\theta|D)}[p(y|x,\theta)]$$

Total uncertainty can be expressed as the entropy of predictive distribution $H(\mathbf{y}|x,D)$. Epistemic uncertainty is estimated as the information gain:

$$I[\mathbf{y},\theta|x,D] = H[\mathbb{E}_{p(\theta|D)}(\mathbf{y}|x,\theta)] - \mathbb{E}_{p(\theta|D)}[H(\mathbf{y}|x,\theta)]$$

Adversarial Attack - Example

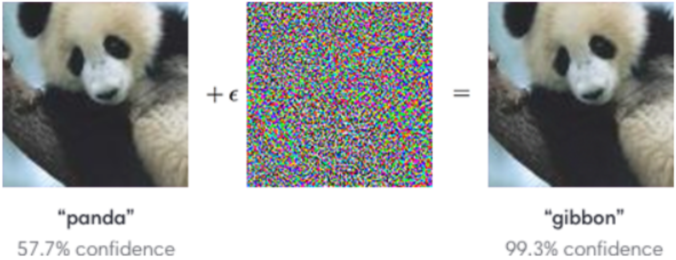


Figure: Source: Open AI Research

Adversarial Attack - Examples MNIST

Source: own work

How can we find adversarial instances?

Given a classifier f, an input instance x and a corresponding label y we have:

$$x' = \arg\max_{z \in S_\epsilon} L_{CE}(f, z, y)$$

where $S_\epsilon = \{z : d(z, x) < \epsilon\}$. This is known as a **white-box** attack since we have direct access to f.

Projected Gradient Descent (PGD) can be used to efficiently solve the above

$$x_{i+1} = \Pi_{S_\epsilon}(x_i + \alpha \nabla_{x_i} L_{CE}(f, x_i, y))$$

where Π_{S_ϵ} denotes a projection onto the set S_ϵ. For instance with $d(z, x) = ||x - z||_\infty$ we project by clipping z to $[x - \epsilon, x + \epsilon]$.

Selective Gradient Descent

Assume we have a scoring function g that detects adversarial examples when $g(x) < 0$. We want to generate an adversarial input with additional constraint. We can use PGD with an augmented loss

$$x' = \arg\max_{z \in S_\epsilon} L_{CE}(f, z, y) + \lambda g(z)$$

but this can lead to *perturbation waste*. **Selective attack** is more efficient

$$x' = \arg\max_{z \in S_\epsilon} L_{CE}(f, z, y)\mathbb{1}(f(z) = y) + \lambda g(z)\mathbb{1}(f(z) \neq y)$$

Adversarial Training - Problem Formulation

Given a dataset D and a parametric model $f(\theta)$ we can formulate standard training procedure

$$\theta' = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim D} \left[L_{CE}(\theta, x, y)\right]$$

In adversarial training, we formulate the following minimax problem:

$$\theta' = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim D} \left[\arg\max_{z \in S_{\epsilon,x}} L_{CE}(\theta, z, y)\right]$$

To solve the inner maximization problem we can again use PGD (PGD-AT)

$$x_{i+1} = \Pi_{S_\epsilon}(x_i + \alpha sign(\nabla_{x_i} L_{CE}(f, x_i, y)))$$

Adversarial Training - Impact on Model

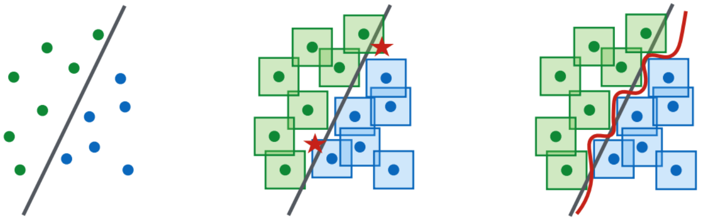


Figure: A conceptual illustration of standard vs. adversarial decision boundaries. Source: Madry, Aleksander, et al. "Towards deep learning models resistant to adversarial attacks."

Adversarial Uncertainty Training

- Adversarial Training makes a model robust in changing its decision
- What if we want to have a model that estimates uncertainty in a robust way?
- Robust uncertainty - insensitive to **non-semantic** changes in input

$$\theta' = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim D}\left[\arg\max_{z\in S_{\epsilon,x}} L_{CE}(\theta, z, y) + \lambda h(\theta, z, x, y)\right]$$

where h regularizes the magnitude of change in probabilities

$$h(\theta, z, x, y) = f(x, \theta)[y](||f(x, \theta) - f(z, \theta)||^2)$$

Research Questions

- Can deep neural networks capable of uncertainty estimation achieve adversarial robustness?
- Does adversarial training hinder the performance of models capable of uncertainty estimation?
- How vulnerable to attacks are decision making systems based on uncertainty estimation?
- Does adversarial uncertainty training improve robustness of decision making systems based on uncertainty estimation?

Dataset - MNIST

Figure: Sample digits from MNIST dataset

Base Model - Lenet 5

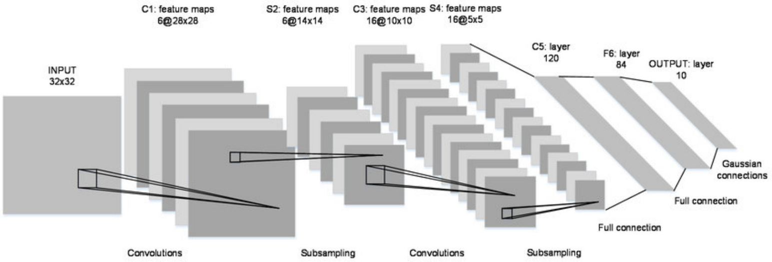
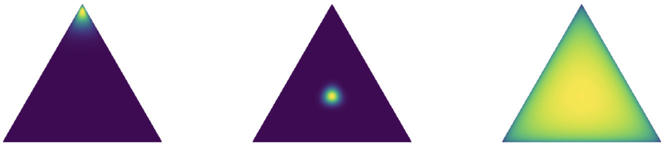


Figure: Tra, Viet, et al. "Bearing fault diagnosis under variable speed using convolutional neural networks and the stochastic diagonal levenberg-marquardt algorithm."

Uncertainty Estimation Methods Used

- Deterministic model trained with CE loss
  - Epistemic uncertainty is 0
- Bayesian Neural Network trained with Variational Inference
  - We approximate the posterior distribution over model weights with a parametric family $p(\theta|D) \approx q_w(\theta)$
  - Optimize w with SGD
- Dirichlet Prior Network
  - Parameterize a Dirichlet distribution - a conjugate prior to the categorical distribution
  - $p(\mu|x) = Dir(\mu;\alpha)$, $\alpha = f(x)$
  - $p(y_c|x) = \int p(y_c|\mu)p(\mu|x)d\mu = \frac{\alpha_c}{\sum_c \alpha_c}$

Dirichlet Distribution and Uncertainty



(a) Confident Prediction (b) High data uncertainty (c) Out-of-distribution

Figure: Source: Malinin, Andrey, and Mark Gales. "Predictive uncertainty estimation via prior networks."

Decision Making System

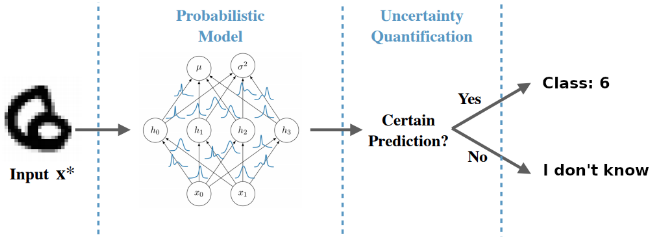


Figure: Workflow of automated decision making system capable of estimating uncertainty and abstaining from giving a decision.

Possible Attacks

- Attack on misclassification detection
  - Maximize uncertainty for correct predictions
  - Minimize uncertainty for incorrect predictions
  - Try not to change predicted class
  - Implemented as a **PGD Attack** and **Selective Attack**
- Attack on the decision making system
  - **Uncertainty Attack** - maximize uncertainty regardless of the prediction
  - **Misclassification Attack** - minimize uncertainty while maintaining an incorrect prediction
- Attack parameters: $\alpha = 0.01$, $k = 40$, $d(x, z) = ||x - z||_\infty$ used in both training and attacking phase.

Adversarial Robustness

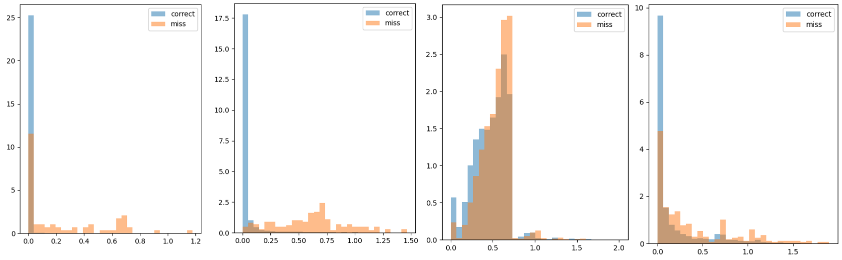
Adversarial robustness against a PGD attack.

Model	Training	Adversarial Accuracy
Prior Lenet 5	Adversarial	0.8737
Prior Lenet 5	Adv Uncertainty	0.9142
Lenet 5	Adversarial	0.9027
Lenet 5	Adv Uncertainty	0.9182
BNN Lenet 5	Adversarial	0.8534
BNN Lenet 5	Adv Uncertainty	0.8863

Misclassification Detection

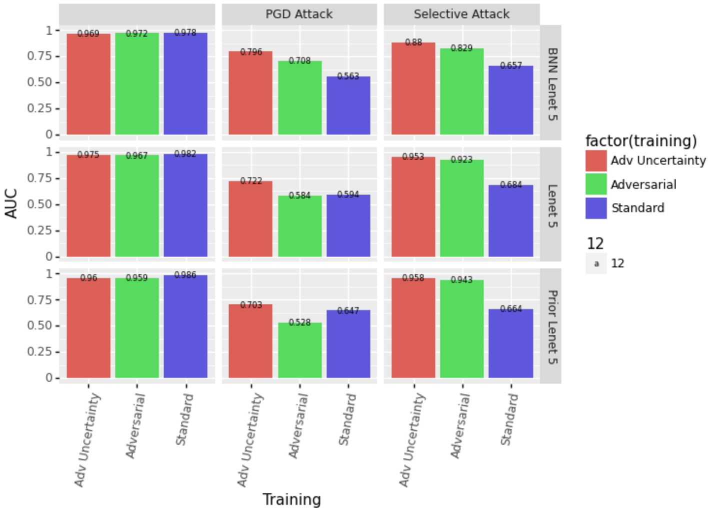
- We can pose misclassification detection as a binary classification problem
- Misclassified instances are of positive class
- Use e.g. prediction uncertainty as the score
- Performance can be measured using Area Under the Roc Curve (AUC)

Misclassification Detection



Density over prediction uncertainty for correct and misclassified inputs. Left side: results on unperturbed inputs. Right side: results using PGD Attack.

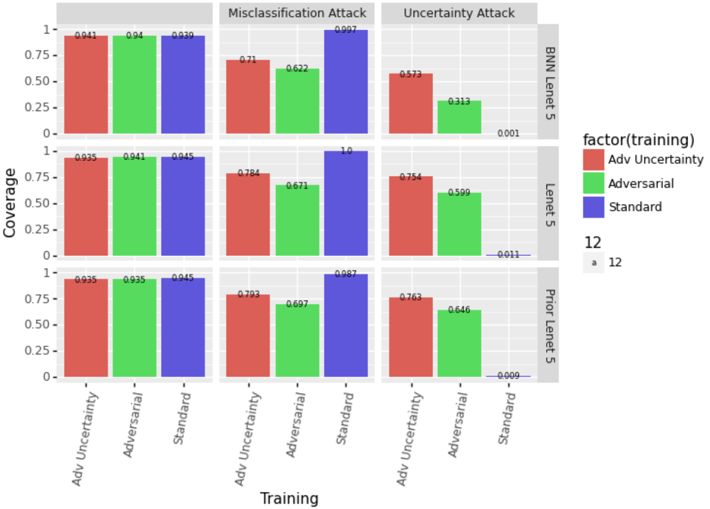
Misclassification Detection



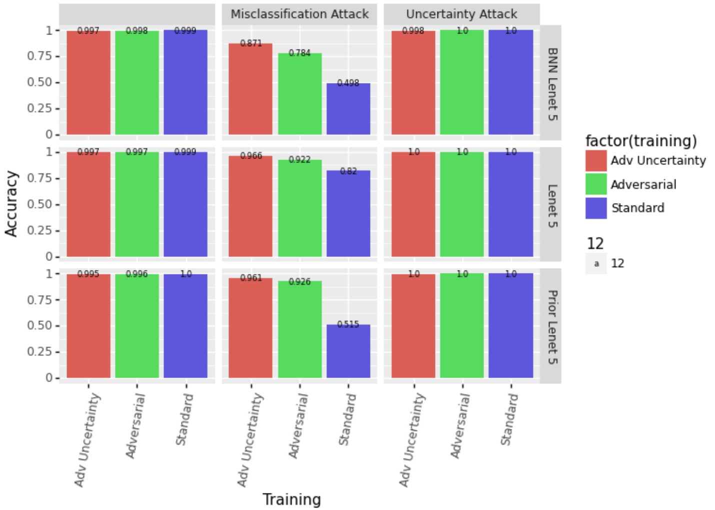
Decision System - Metrics

- **Coverage** - percentage of instances for which the model returned a prediction
- **Accuracy** - classification accuracy on covered instances
- How is the uncertainty decision threshold estimated?
  - Select a threshold s.t. the misclassification detection false positive rate is 5%

Decision System Robustness - Coverage



Decision System Robustness - Accuracy



Thank you