Adversarial Uncertainty Learning in Deep Neural Networks

Łukasz Grad

University of Warsaw

l.grad@uw.edu.pl

October 29, 2021
Overview

1. Uncertainty in Probabilistic Modelling
   - Kinds of Uncertainty
   - Uncertainty Measures

2. Adversarial Machine Learning
   - Adversarial Attacks
   - Adversarial Learning

3. Experimental Setup
   - Research Questions
   - Dataset and Models Used
   - Decision System and Attacks

4. Results
   - Adversarial Robustness
   - Misclassification Detection
   - Decision System Robustness
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 in Probabilistic Modelling
Adversarial Machine Learning
Experimental Setup
Results

Kinds of Uncertainty
Uncertainty Measures

Uncertainty - Regression

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

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

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 $H(y|x, D)$. Epistemic uncertainty is estimated as the information gain:

$$I[y, \theta|x, D] = H[\mathbb{E}_{p(\theta|D)}(y|x, \theta)] - \mathbb{E}_{p(\theta|D)}[H(y|x, \theta)]$$
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 $\Pi_{S_\epsilon}$ denotes a projection onto the set $S_\epsilon$. For instance with $d(z, x) = \|x - z\|_\infty$ we project by clipping $z$ to $[x - \epsilon, x + \epsilon]$. 
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

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

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)

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

<table>
<thead>
<tr>
<th>Model</th>
<th>Training</th>
<th>Adversarial Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prior Lenet 5</td>
<td>Adversarial</td>
<td>0.8737</td>
</tr>
<tr>
<td>Prior Lenet 5</td>
<td>Adv Uncertainty</td>
<td>0.9142</td>
</tr>
<tr>
<td>Lenet 5</td>
<td>Adversarial</td>
<td>0.9027</td>
</tr>
<tr>
<td>Lenet 5</td>
<td>Adv Uncertainty</td>
<td>0.9182</td>
</tr>
<tr>
<td>BNN Lenet 5</td>
<td>Adversarial</td>
<td>0.8534</td>
</tr>
<tr>
<td>BNN Lenet 5</td>
<td>Adv Uncertainty</td>
<td>0.8863</td>
</tr>
</tbody>
</table>
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).
Density over prediction uncertainty for correct and misclassified inputs. Left side: results on unperturbed inputs. Right side: results using PGD Attack.
Misclassification Detection

![Graph showing AUC for different training methods and attack types]

- **Training Methods:**
  - Adv Uncertainty
  - Adversarial
  - Standard

- **Attack Types:**
  - PGD Attack
  - Selective Attack

- **Models:**
  - BNN Lenet 5
  - Lenet 5
  - Prior Lenet 5

- **Factor:**
  - (training)

- **Legend:**
  - Adv Uncertainty
  - Adversarial
  - Standard

- **AUC Values:**
  - 0.969, 0.972, 0.978
  - 0.975, 0.967, 0.982
  - 0.96, 0.959, 0.986
  - 0.976, 0.708, 0.563
  - 0.88, 0.829, 0.657
  - 0.953, 0.923, 0.684

- **Values:**
  - 0.958, 0.943, 0.664

- **Indices:**
  - 12
  - a 12
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

![Bar charts showing coverage for different training types and models: BNN, Lenet 5, Lenet 5, Prior Lenet 5. The x-axis represents training (Adv Uncertainty, Adversarial, Standard), and the y-axis represents coverage. The charts compare adversarial uncertainty, adversarial, and standard training methods.](image)
Decision System Robustness - Accuracy

![Bar chart showing accuracy for different training types](image)

- **Training Types**: Adv Uncertainty, Adversarial, Standard
- **Networks**: BNN Lenet 5, Lenet 5, Prior Lenet 5
- **Factors (training)**: 12

The chart illustrates the accuracy of decision system robustness under various training conditions for different network architectures.
Thank you