What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision

Patryk Chrabąszcz
Outline:

● Motivation
● Types of Uncertainty
● Bayesian Neural Networks
● Dropout Variational Inference
● Modeling uncertainties
● Experiments
● Results Analysis
● Summary
Importance of modeling uncertainty
Importance of modeling uncertainty

- Autonomous Car Accident
Importance of modeling uncertainty

- Autonomous Car Accident
- Google app racial discrimination
Importance of modeling uncertainty

- Autonomous Car Accident
- Google app racial discrimination
- Safety Critical Systems
Importance of modeling uncertainty

- Autonomous Car Accident
- Google app racial discrimination
- Safety Critical Systems
- Medical Applications

I have never seen those symptoms before. I’m completely uncertain what it could be. Better see a doctor!
Example: Active Learning
Example: Active Learning

- Model decides which data should be labeled
Example: Active Learning

- Model decides which data should be labeled

Source: “Cost-Effective Active Learning for Deep Image Classification”
Example: Active Learning

- Model decides which data should be labeled
- Collect the best data at low cost

Source: “Cost-Effective Active Learning for Deep Image Classification”
Example: Reinforcement Learning
Example: Reinforcement Learning

- No knowledge at the start
Example: Reinforcement Learning

- No knowledge at the start
- Make decision at each step
  - Explore ?
  - Exploit ?
Example: Reinforcement Learning

● No knowledge at the start
● Make decision each step
  ○ Explore ?
  ○ Exploit ?
● Intelligent exploration
Outline:

- Motivation
- **Types of Uncertainty**
- Bayesian Neural Networks
- Dropout Variational Inference
- Modeling uncertainties
- Experiments
- Results Analysis
- Summary
Uncertainty

- Aleatoric
  - Homoscedastic
- Epistemic
  - Heteroscedastic
Aleatoric uncertainty

- Natural randomness

Modeling the result of dice throw.
In latin ālea: a die
Aleatoric uncertainty

- Natural randomness
- Sensor quality
Aleatoric uncertainty

- Natural randomness
- Sensor quality
- Can’t be reduced
Aleatoric uncertainty

- Natural randomness
- Sensor quality
- Can’t be reduced
- But can be learned
Uncertainty

- Aleatoric
  - Homoscedastic
- Epistemic
  - Heteroscedastic
Homoscedastic uncertainty

- Stays constant for different input values

Homoscedastic uncertainty

- Stays constant for different input values
- Limited, captures ‘average’ uncertainty

Uncertainty

Aleatoric

Homoscedastic

Epistemic

Heteroscedastic
Heteroscedastic uncertainty

- Depends on the input

Heteroscedastic uncertainty

- Depends on the input
- Important for CV tasks

Heteroscedastic uncertainty

- Depends on the input
- Important for CV tasks
- Learned from the data

Uncertainty

Aleatoric
- Homoscedastic

Epistemic
- Heteroscedastic
Epistemic uncertainty

- Lack of knowledge about the process

*Epistēmē* Greek meaning: *knowledge.*
Epistemic uncertainty

- Lack of knowledge about the process
- Detects samples far from the training distribution
Epistemic uncertainty

- Lack of knowledge about the process
- Detects samples far from training distribution
- Disappears given enough data.
Epistemic uncertainty

- Lack of knowledge about the process
- Detects samples far from training distribution
- Disappears given enough data.
- **Train many models, detect where models disagree**
Epistemic uncertainty

- Lack of knowledge about the process
-Detects samples far from training distribution
- Disappears given enough data.
- Train many models, detect where models disagree
- **Use distribution over model weights**
Outline:

- Motivation
- Types of Uncertainty
- **Bayesian Neural Networks**
- Dropout Variational Inference
- Modeling uncertainties
- Experiments
- Results Analysis
- Summary
Bayesian Neural Networks

- Neural Network
Bayesian Neural Networks

- **Neural Network**
  - Finds a function $y = f(x)$
Bayesian Neural Networks

- Neural Network
  - Finds a function $y = f(x)$
  - One best model
Bayesian Neural Networks

- Neural Network
  - Finds a function $y = f(x)$
  - One best model
- Bayesian Neural Network
Bayesian Neural Networks

- **Neural Network**
  - Finds a function $y = f(x)$
  - One best model

- **Bayesian Neural Network**
  - Distribution over weights
Bayesian Neural Networks

- Neural Network
  - Finds a function $y = f(x)$
  - One best model

- Bayesian Neural Network
  - Distribution over weights
  - Output is a distribution
Bayesian Neural Networks

- Neural Network
  - Finds a function $y = f(x)$
  - One best model

- Bayesian Neural Network
  - Distribution over weights
  - Output is a distribution
  - Many models within a network
Bayesian Neural Networks
Bayesian Neural Networks

- Ideas from 30 years ago

Source: Gal, Y.  
http://mlg.eng.cam.ac.uk/yarin/index.html
Bayesian Neural Networks

- Ideas from 30 years ago
- **BNN with $\infty$ many weights $\rightarrow$ Gaussian Process**
Bayesian Neural Networks

- Ideas from 30 years ago
- BNN with $\infty$ many weights $\rightarrow$ Gaussian Process
- **Difficult to make inference on:**
  - Multimodal correlated distribution
  - Nonlinearities

Source: Gal, Y.  
http://mlg.eng.cam.ac.uk/yarin/index.html
Bayesian Neural Networks

- Ideas from 30 years ago
- BNN with $\infty$ many weights $\rightarrow$ Gaussian Process
- Difficult to make inference on:
  - Multimodal correlated distribution
  - Nonlinearities
- Different approximation techniques

Source: Gal, Y. http://mlg.eng.cam.ac.uk/yarin/index.html
Bayesian Neural Networks

- Find $p(W|X, Y)$
Bayesian Neural Networks

- Find $p(W|X, Y)$
  - Find models that are likely to generate our training data
Bayesian Neural Networks

- **Find** $p(W|X, Y)$

  - Find models that are likely to generate our training data

$$p(W|X, Y) = \frac{p(Y|X, W)p(W)}{p(Y|X)}$$
Bayesian Neural Networks

- Likelihood (Gaussian, Laplace)
  - Measures how likely model with weights $W$ generated $Y$

$$p(W|X, Y) = \frac{p(Y|X, W)p(W)}{p(Y|X)}$$
Bayesian Neural Networks

- Likelihood (Gaussian, Laplace)
  - Measures how likely model with weights $W$ generated $Y$

\[
p(y | f^W(x)) = \mathcal{N}(f^W(x), \sigma^2)
\]

\[
p(W | X, Y) = \frac{p(Y | X, W)p(W)}{p(Y | X)}
\]
Bayesian Neural Networks

- Prior
  - Usually a Gaussian distribution with mean at 0

\[
p(W|X, Y) = \frac{p(Y|X, W)p(W)}{p(Y|X)}
\]
Bayesian Neural Networks

● Prior
  ○ Usually a Gaussian distribution with mean at 0
  ○ Acts as a regularizer

\[
p(W|X, Y) = \frac{p(Y|X, W)p(W)}{p(Y|X)}
\]
Bayesian Neural Networks

- **Marginal Probability**
  - Normalizes probability

\[
p(W|X, Y) = \frac{p(Y|X, W)p(W)}{p(Y|X)}
\]
Bayesian Neural Networks

- **Marginal Probability**
  - Normalizes probability
  - Can not be evaluated

\[
p(W | X, Y) = \frac{p(Y | X, W)p(W)}{p(Y | X)} \int p(Y | X, W)p(W) dW
\]
Bayesian Neural Networks

- In general $p(W|X, Y)$ can be a complex distribution
Bayesian Neural Networks

- In general, $p(W|X, Y)$ can be a complex distribution
- We need an approximation
Bayesian Neural Networks

- In general, $p(W|X, Y)$ can be a complex distribution.
- We need an approximation.
- Replace complex $p(W|X, Y)$ with $q^{*}_{\theta}(W)$ from a tractable family (Gaussian, Bernoulli).
Bayesian Neural Networks

- In general, $p(W|X, Y)$ can be a complex distribution.
- We need an approximation.
- Approximate complex $p(W|X, Y)$ with $q^*_\theta(W)$ from a tractable family (Gaussian, Bernoulli).
- Minimize the distance between them (KL Divergence).
Outline:

● Motivation
● Types of Uncertainty
● Bayesian Neural Networks
● **Dropout Variational Inference**
● Modeling uncertainties
● Experiments
● Results Analysis
● Summary
Dropout Variational Inference

- Randomly drop network units

Source: “Dropout: A Simple Way to Prevent Neural Networks from Overfitting” Srivastava et. al.
Dropout Variational Inference

- Randomly drop network units
- Bernoulli approximation $q_\theta^*(W)$

Source: “Dropout: A Simple Way to Prevent Neural Networks from Overfitting” Srivastava et. al.
Dropout Variational Inference

- Randomly drop network units
- Bernoulli approximation $q_\theta^*(W)$
- One of the simplest possible

Source: “Dropout: A Simple Way to Prevent Neural Networks from Overfitting” Srivastava et. al.
Dropout Variational Inference

- Randomly drop network units
- Bernoulli approximation \( q^*_\theta(W) \)
- One of the simplest possible
- **We learn \( \theta \) for each weight**

Source: “Dropout: A Simple Way to Prevent Neural Networks from Overfitting” Srivastava et. al.
Training with dropout

- Neural network trained with dropout is already a BNN
Training with dropout

- Neural network trained with dropout is already a BNN
  - Because we have a distribution over its weights
Training with dropout

- Neural network trained with dropout is already a BNN

\[
L(\theta, p) = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i|f^{\hat{W}_i}(x_i)) + \frac{1 - p}{2N} ||\theta||^2
\]
Training with dropout

- Neural network trained with dropout is already a BNN

\[ L(\theta, p) = - \frac{1}{N} \sum_{i=1}^{N} \log p(y_i | f^{\hat{W}_i}(x_i)) + \frac{1 - p}{2N} ||\theta||^2 \]

For each training point
draw a new dropout mask
Training with dropout

- Neural network trained with dropout is already a BNN

\[
L(\theta, p) = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i | f_{\hat{W}_i}(x_i)) + \frac{1 - p}{2N} \|\theta\|^2
\]

Minimize negative log likelihood
Training with dropout

- Neural network trained with dropout is already a BNN

\[ L(\theta, p) = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i | f_{\hat{W}_i}(x_i)) + \frac{1 - p}{2N} \|\theta\|^2 \]

Regularization term (prior)
Training with dropout

- Neural network trained with dropout is already a BNN

\[ L(\theta, p) = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i | f^{\hat{W}_i}(x_i)) + \frac{1 - p}{2N} ||\theta||^2 \]

- In this work dropout probability \( p \) is set to 0.2
Training with dropout

● Neural network trained with dropout is already a BNN

\[ L(\theta, p) = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i | f_{\hat{\theta}}^i(x_i)) + \frac{1 - p}{2N} ||\theta||^2 \]

● In this work dropout probability \( p \) is set to 0.2

● Minimizing this loss we also minimize KL divergence between \( p(W | X, Y) \) and \( q_{\theta}^*(W) \)
Outline:

- Motivation
- Types of Uncertainty
- Bayesian Neural Networks
- Dropout Variational Inference
- **Modeling uncertainties**
- Experiments
- Results Analysis
- Summary
Modeling uncertainties

Uncertainty

Aleatoric

Homoscedastic

Epistemic

Heteroscedastic
Homoscedastic uncertainty

- Regression Problem
Homoscedastic uncertainty

- Regression Problem

\[ L(\theta, p) = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i | f_{\hat{W}_i}(x_i)) + \frac{1-p}{2N} ||\theta||^2 \]
Homoscedastic uncertainty

- Regression Problem

\[ L(\theta, p) = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i \mid f \hat{W}_i(x_i)) + \frac{1 - p}{2N} \|\theta\|^2 \]

- If Laplace Likelihood:

\[ \frac{1}{\sigma^2} \|y_i - f \hat{W}_i(x_i)\| + \log \sigma^2 \]
Homoscedastic uncertainty

● Regression Problem

\[ L(\theta, p) = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i \mid f\hat{W}_i(x_i)) + \frac{1 - p}{2N} \|\theta\|^2 \]

● If Laplace Likelihood:
  ○ Minimize the distance between model predictions and the training data

\[ \frac{1}{\sigma^2} \| y_i - f\hat{W}_i(x_i) \| + \log \sigma^2 \]
HOMOSCEDASTIC UNCERTAINTY

- Regression Problem

\[
L(\theta, p) = \frac{-1}{N} \sum_{i=1}^{N} \log p(y_i | f_{\hat{W}_i}(x_i)) + \frac{1 - p}{2N} ||\theta||^2
\]

- If Laplace Likelihood

\[
\frac{1}{\sigma^2} ||y_i - f_{\hat{W}_i}(x_i)|| + \log \sigma^2
\]

- Use sigma and dropout samples to estimate uncertainty
Estimated variance

- Sum of aleatoric and epistemic variance
Estimated variance

- Sum of aleatoric and epistemic variance
  - Epistemic variance: variance over multiple dropout draws

\[
Var(y) \approx \sigma^2 + \frac{1}{T} \sum_{t=1}^{T} (f^{\hat{W}_t}(x) - E(y))^2
\]

\[
E(y) \approx \frac{1}{T} \sum_{t=1}^{T} f^{\hat{W}_t}(x)
\]
Heteroscedastic uncertainty

Uncertainty

Aleatoric

Homoscedastic

Epistemic

Heteroscedastic
Heteroscedastic uncertainty

- Start as before

\[ \frac{1}{\sigma^2} \| y_i - f \hat{W}_i (x_i) \| + \log \sigma^2 \]
Heteroscedastic uncertainty

- Start as before

$\frac{1}{\sigma^2} \left\| y_i - f(\hat{W}_i(x_i)) \right\| + \log \sigma^2$

- Uncertainty as a function of the input

$\frac{1}{\sigma(x_i)^2} \left\| y_i - f(\hat{W}_i(x_i)) \right\| + \log \sigma(x_i)^2$
Heteroscedastic uncertainty

- Start as before

\[ \frac{1}{\sigma^2} \| y_i - f(\hat{W}_i(x_i)) \| + \log \sigma^2 \]

- Uncertainty as a function of the input
  - If it is hard to predict correct output, increase uncertainty to reduce loss
  - Uncertainty acts as a loss attenuation
  - Robust to outliers
Heteroscedastic uncertainty

- Start as before

- Uncertainty as a function of the input

\[
\frac{1}{\sigma^2} \| y_i - f(\hat{W}_i(x_i)) \| + \log \sigma^2
\]

\[
\frac{1}{\sigma(x_i)^2} \| y_i - f(\hat{W}_i(x_i)) \| + \log \sigma(x_i)^2
\]
Classification with Uncertainty

- Modeling uncertainty for classification
Classification with Uncertainty

- Modeling uncertainty for classification
  - Add noise to the output of the network (logits)
Classification with Uncertainty

- Modeling uncertainty for classification
  - Add noise to the output of the network (logits)
  - Variance of this noise depends on the input
Classification with Uncertainty

- Modeling uncertainty for classification
  - Add noise to the output of the network (logits)
  - Variance of this noise depends on the input

\[
\hat{x}_{i,t} = f_i^W + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, (\sigma_i^W)^2)
\]

\[
\mathcal{L}_x = \frac{1}{T} \sum_{i,t} (-\hat{x}_{i,t,c} + \log \sum_{c'} \exp \hat{x}_{i,t,c'})
\]
Classification with Uncertainty

- Modeling uncertainty for classification
  - Add noise to the output of the network (logits)
  - Variance of this noise depends on the input

\[ \hat{x}_{i,t} = f_i^W + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, (\sigma_i^W)^2) \]
\[ \mathcal{L}_x = \frac{1}{T} \sum_{i,t} (-\hat{x}_{i,t,c} + \log \sum_{c'} \exp \hat{x}_{i,t,c'}) \]

- If model is wrong, bigger uncertainty results in a lower loss
Classification with Uncertainty

- Example
Classification with Uncertainty

Network outputs:
First class: 1
Second class: 2
Classification with Uncertainty

After softmax:
First class: 27%
Second class: 73%
Classification with Uncertainty

Sample 50 times logits with noise (variance = 0.5)

27%
73%
Classification with Uncertainty

27%  73%

30%  70%
Outline:

- Motivation
- Types of Uncertainty
- Bayesian Neural Networks
- Dropout Variational Inference
- Modeling uncertainties
- Experiments
- Results Analysis
- Summary
CamVid

- Road scene understanding dataset
- 367 train, 233 test
- Day and dusk scenes
- 11 classes
- Resized to 360×480
NYUv2 40-Class

- Indoor segmentation dataset
- 40 different semantic classes
- 464 different indoor scenes.
- 1449 images
- 640×480
NYUv2 Depth

● Indoor dataset
● 464 different indoor scenes.
● 1449 images
● 640×480
Make 3D

- 400 training, 134 test
- 3-D laser scanner.
- Resized to 345×460
Outline:

- Motivation
- Types of Uncertainty
- Bayesian Neural Networks
- Dropout Variational Inference
- Modeling uncertainties
- Experiments
- Results Analysis
- Summary
Semantic Segmentation

Aleatoric

Epistemic
Semantic Segmentation

Aleatoric

Epistemic
Pixel-wise Depth Regression

Aleatoric

Epistemic
Pixel-wise Depth Regression

Aleatoric

Epistemic
Quality of Uncertainty Metric

- Uncertainty correlates with accuracy
Quality of Uncertainty Metric

- Uncertainty correlates with accuracy
- Precision decreases as we increase uncertainty
Calibration

- If prediction has a probability of “p”, we would like this prediction to be correct with frequency “p”
Calibration

- If prediction has a probability of “p”, we would like this prediction to be correct with frequency “p”
Dataset size

- Modeling epistemic variance should be more beneficial when our training data is small
Dataset size

- Modeling epistemic variance should be more beneficial when our training data is small
- Epistemic variance captures data from different distribution

<table>
<thead>
<tr>
<th>Train dataset</th>
<th>Test dataset</th>
<th>RMS</th>
<th>Aleatoric variance</th>
<th>Epistemic variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make3D / 4</td>
<td>Make3D</td>
<td>5.76</td>
<td>0.506</td>
<td>7.73</td>
</tr>
<tr>
<td>Make3D / 2</td>
<td>Make3D</td>
<td>4.62</td>
<td>0.521</td>
<td>4.38</td>
</tr>
<tr>
<td>Make3D</td>
<td>Make3D</td>
<td>3.87</td>
<td>0.485</td>
<td>2.78</td>
</tr>
<tr>
<td>Make3D / 4</td>
<td>NYUv2</td>
<td>-</td>
<td>0.388</td>
<td>15.0</td>
</tr>
<tr>
<td>Make3D</td>
<td>NYUv2</td>
<td>-</td>
<td>0.461</td>
<td>4.87</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Train dataset</th>
<th>Test dataset</th>
<th>IoU</th>
<th>Aleatoric entropy</th>
<th>Epistemic logit variance ($\times 10^{-3}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CamVid / 4</td>
<td>CamVid</td>
<td>57.2</td>
<td>0.106</td>
<td>1.96</td>
</tr>
<tr>
<td>CamVid / 2</td>
<td>CamVid</td>
<td>62.9</td>
<td>0.156</td>
<td>1.66</td>
</tr>
<tr>
<td>CamVid</td>
<td>CamVid</td>
<td>67.5</td>
<td>0.111</td>
<td>1.36</td>
</tr>
<tr>
<td>CamVid / 4</td>
<td>NYUv2</td>
<td>-</td>
<td>0.247</td>
<td>10.9</td>
</tr>
<tr>
<td>CamVid</td>
<td>NYUv2</td>
<td>-</td>
<td>0.264</td>
<td>11.8</td>
</tr>
</tbody>
</table>
# Improvement over Baseline

<table>
<thead>
<tr>
<th>CamVid Results</th>
<th>IoU Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>DenseNet (State of the art baseline)</td>
<td>67.1</td>
</tr>
<tr>
<td>+ Aleatoric Uncertainty</td>
<td>67.4</td>
</tr>
<tr>
<td>+ Epistemic Uncertainty</td>
<td>67.2</td>
</tr>
<tr>
<td>+ Aleatoric &amp; Epistemic</td>
<td>67.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NYU Depth Results</th>
<th>Rel. Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>DenseNet (State of the art baseline)</td>
<td>0.167</td>
</tr>
<tr>
<td>+ Aleatoric Uncertainty</td>
<td>0.149</td>
</tr>
<tr>
<td>+ Epistemic Uncertainty</td>
<td>0.162</td>
</tr>
<tr>
<td>+ Aleatoric &amp; Epistemic</td>
<td>0.145</td>
</tr>
</tbody>
</table>

Source: http://alexgkendall.com/talks/
Summary:

● Aleatoric uncertainty:
  ○ Can be used for real time applications
  ○ Can be used alone for large datasets

● Epistemic uncertainty:
  ○ Can detect samples out of the training data
  ○ Useful for small datasets
  ○ Expensive to evaluate (MC Sampling)
Thank you