Bayesian Deep Learning

Yarin Gal

Research Fellow, University of Cambridge
Research Fellow, The Alan Turing Institute
yg279@cam.ac.uk

Unless specified otherwise, photos are either original work or taken from Wikimedia, under Creative Commons license
Pillar I: Deep learning

Conceptually simple models

Data: $X = \{x_1, x_2, \ldots, x_N\}$, $Y = \{y_1, y_2, \ldots, y_N\}$

Model: given matrices $W$ and non-linear func. $\sigma(\cdot)$, define “network”

$$\hat{y}_i(x_i) = W_2 \cdot \sigma(W_1 x_i)$$

Objective: find $W$ for which $\hat{y}_i(x_i)$ is close to $y_i$ for all $i \leq N$. 
Pillar I: Deep learning

Conceptually simple models

**Data:** $X = \{x_1, x_2, \ldots, x_N\}$, $Y = \{y_1, y_2, \ldots, y_N\}$

**Model:** given matrices $W$ and non-linear func. $\sigma(\cdot)$, define “network”

$$\hat{y}_i(x_i) = W_2 \cdot \sigma(W_1 x_i)$$

**Objective:** find $W$ for which $\hat{y}_i(x_i)$ is close to $y_i$ for all $i \leq N$.

Deep learning is awesome ✔️ ... but has many issues ❌

- Simple and modular
- Huge attention from practitioners and engineers
- Great software tools
- Scales with data and compute
- Real-world impact

- What does a model not know?
- Uninterpretable black-boxes
- Easily fooled (AI safety)
- Lacks solid mathematical foundations (mostly ad hoc)
- Crucially relies on big data
Pillar I: Deep learning

Conceptually simple models

Data: $X = \{x_1, x_2, \ldots, x_N\}$, $Y = \{y_1, y_2, \ldots, y_N\}$

Model: given matrices $W$ and non-linear func. $\sigma(\cdot)$, define “network”

$$\hat{y}_i(x_i) = W_2 \cdot \sigma(W_1 x_i)$$

Objective: find $W$ for which $\hat{y}_i(x_i)$ is close to $y_i$ for all $i \leq N$.

Deep learning is awesome ✔️

- Simple and modular
- Huge attention from practitioners and engineers
- Great software tools
- Scales with data and compute
- Real-world impact

... but has many issues ✗

- What does a model not know?
- Uninterpretable black-boxes
- Easily fooled (AI safety)
- Lacks solid mathematical foundations (mostly ad hoc)
- Crucially relies on big data
Why should I care about uncertainty?

- We need a way to tell **what our model knows** and what not.
  - We train a model to recognise dog breeds

---

Uncertainty gives insights into the black-box when it fails — where am I not certain?

Uncertainty might even be useful to identify when attacked with adversarial examples!

Lastly, need less data if label only where model is uncertain: wear-and-tear in robotics, expert time in medical analysis.
Why should I care about uncertainty?

- We need a way to tell **what our model knows** and what not.
  - We train a model to recognise dog breeds
  - And are given a cat to classify

Uncertainty gives insights into the black-box when it fails — where am I not certain?

Uncertainty might even be useful to identify when attacked with adversarial examples!

Lastly, need less data if label only where model is uncertain: wear-and-tear in robotics, expert time in medical analysis.
Why should I care about uncertainty?

- We need a way to tell what our model knows and what not.
  - We train a model to recognise dog breeds
  - And are given a cat to classify
  - What would you want your model to do?

Uncertainty gives insights into the black-box when it fails — where am I not certain?

Uncertainty might even be useful to identify when attacked with adversarial examples!

Lastly, need less data if label only where model is uncertain: wear-and-tear in robotics, expert time in medical analysis.
Why should I care about uncertainty?

- We need a way to tell what our model knows and what not.
  - We train a model to recognise dog breeds
  - And are given a cat to classify
  - What would you want your model to do?
  - Similar problems in decision making, physics, life science, etc.
Why should I care about uncertainty?

► We need a way to tell **what our model knows** and what not.

► Uncertainty gives insights into the black-box when it fails —where am I not certain?
Why should I care about uncertainty?

- We need a way to tell **what our model knows** and what not.
- Uncertainty gives insights into the black-box when it fails —where am I not certain?
- Uncertainty might even be useful to identify when attacked with adversarial examples!

Lastly, need less data if label only where **model is uncertain**: wear-and-tear in robotics, expert time in medical analysis.
Why should I care about uncertainty?

- We need a way to tell **what our model knows** and what not.
- Uncertainty gives insights into the black-box when it fails —where am I not certain?
- Uncertainty might even be useful to identify when attacked with adversarial examples!
- Lastly, need less data if label only where **model is uncertain**: wear-and-tear in robotics, expert time in medical analysis.
Pillar II: Bayes

The language of uncertainty

- Probability theory
- Specifically *Bayesian probability theory* (1750!)

When applied to *Information Engineering*...

- Bayesian modelling

- Built on solid mathematical foundations
- Orthogonal to deep learning...
A simple way to tie the two pillars together

- “Dropout”: a popular method in deep learning, cited hundreds and hundreds of times
  - Works by randomly setting network units to zero
  - This *somehow* improves performance and reduces over-fitting
  - Used in almost all modern deep learning models
A simple way to tie the two pillars together

- “Dropout”: a popular method in deep learning, cited hundreds and hundreds of times
- Works by randomly setting network units to zero
- This somehow improves performance and reduces over-fitting
- Used in almost all modern deep learning models
A simple way to tie the two pillars together

▶ “Dropout”: a popular method in deep learning, cited hundreds and hundreds of times

▶ Works by randomly setting network units to zero

▶ This *somehow* improves performance and reduces over-fitting

▶ Used in almost all modern deep learning models
A simple way to tie the two pillars together

- “Dropout”: a popular method in deep learning, cited hundreds and hundreds of times
- Works by randomly setting network units to zero
- This *somehow* improves performance and reduces over-fitting
- Used in almost all modern deep learning models
A simple way to tie the two pillars together

- Can be shown that dropout training is identical to approximate inference in Bayesian modelling [Gal, 2016],

- Connecting Deep Learning to Bayesian probability theory.

- The mathematically grounded connection gives a treasure trove of new research opportunities:
  - uncertainty in deep learning, e.g. interpretability and AI safety
  - principled extensions to deep learning
  - enable deep learning in small data domains
A simple way to tie the two pillars together

- Can be shown that dropout training is identical to *approximate inference in Bayesian modelling* [Gal, 2016],

- Connecting **Deep Learning** to Bayesian probability theory.

- The **mathematically grounded** connection gives a treasure trove of new research opportunities:
  - uncertainty in deep learning, e.g. interpretability and AI safety
  - principled extensions to deep learning
  - enable deep learning in small data domains
A simple way to tie the two pillars together

- Can be shown that dropout training is identical to *approximate inference in Bayesian modelling* [Gal, 2016],

- Connecting **Deep Learning to Bayesian probability theory**.

- The *mathematically grounded* connection gives a treasure trove of new research opportunities:
  - uncertainty in deep learning, e.g. interpretability and AI safety
  - principled extensions to deep learning
  - enable deep learning in small data domains
A simple way to tie the two pillars together

- Can be shown that dropout training is identical to *approximate inference in Bayesian modelling* [Gal, 2016],

- Connecting **Deep Learning to Bayesian probability theory**.

- The **mathematically grounded** connection gives a treasure trove of new research opportunities:
  - **uncertainty** in deep learning, e.g. interpretability and AI safety
  - **principled extensions** to deep learning
  - enable deep learning in **small data** domains
A simple way to tie the two pillars together

- Can be shown that dropout training is identical to \textit{approximate inference in Bayesian modelling} [Gal, 2016],

- Connecting \textbf{Deep Learning to Bayesian probability theory}.

- The \textbf{mathematically grounded} connection gives a treasure trove of new research opportunities:
  - \textbf{uncertainty} in deep learning, e.g. interpretability and AI safety
  - \textbf{principled extensions} to deep learning
  - enable deep learning in \textbf{small data} domains
A simple way to tie the two pillars together

- Can be shown that dropout training is identical to \textit{approximate inference in Bayesian modelling} [Gal, 2016],

- Connecting \textbf{Deep Learning to Bayesian probability theory}.

- The \textbf{mathematically grounded} connection gives a treasure trove of new research opportunities:
  - \textbf{uncertainty} in deep learning, e.g. interpretability and AI safety
  - \textbf{principled extensions} to deep learning
  - enable deep learning in \textbf{small data domains}
A simple way to tie the two pillars together

- Can be shown that dropout training is identical to approximate inference in Bayesian modelling [Gal, 2016],

- Connecting **Deep Learning to Bayesian probability theory**.

- The **mathematically grounded** connection gives a treasure trove of new research opportunities:
  - **uncertainty** in deep learning, e.g. interpretability and AI safety
  - **principled extensions** to deep learning
  - enable deep learning in **small data** domains
Can be shown that dropout training is identical to *approximate inference in Bayesian modelling* [Gal, 2016],

Connecting **Deep Learning to Bayesian probability theory**.

The **mathematically grounded** connection gives a treasure trove of new research opportunities:

- **uncertainty** in deep learning, e.g. interpretability and AI safety
- **principled extensions** to deep learning
- enable deep learning in **small data** domains

More in a second. First, some **theory**.
Some theory

From Bayesian neural networks to Dropout

- Place prior $p(W)$ dist. on weights, making these r.v.s

- Given dataset $X, Y$, the r.v. $W$ has a posterior: $p(W|X, Y)$
From Bayesian neural networks to Dropout

- Place prior $p(W)$ dist. on weights, making these r.v.s

- Given dataset $X, Y$, the r.v. $W$ has a posterior: $p(W|X, Y)$

- Which is difficult to evaluate—many great researchers tried

- Can define simple distribution $q_M(\cdot)$ and approximate

$$q_M(W) \approx p(W|X, Y)$$

- This is called approximate variational inference.
Some theory

From Bayesian neural networks to Dropout

- Place prior $p(W)$ dist. on weights, making these r.v.s

- Given dataset $X, Y$, the r.v. $W$ has a posterior: $p(W|X, Y)$

- Which is difficult to evaluate—many great researchers tried

- Can define simple distribution $q_M(\cdot)$ and approximate
  
  $q_M(W) \approx p(W|X, Y)$

- This is called approximate variational inference.
Some theory

From Bayesian neural networks to Dropout

- Place prior $p(W)$ dist. on weights, making these r.v.s
- Given dataset $X, Y$, the r.v. $W$ has a posterior: $p(W|X, Y)$
- Which is difficult to evaluate—many great researchers tried
- Can define simple distribution $q_M(\cdot)$ and approximate
  $$q_M(W) \approx p(W|X, Y)$$
Some theory

From Bayesian neural networks to Dropout

- Place prior $p(W)$ dist. on weights, making these r.v.s
- Given dataset $X$, $Y$, the r.v. $W$ has a posterior: $p(W|X, Y)$
- Which is difficult to evaluate—many great researchers tried
- Can define simple distribution $q_M(\cdot)$ and approximate
  
  $q_M(W) \approx p(W|X, Y)$

---

**Legend:**
- $q_{\theta_2}(W)$
- $p(W|X, Y)$
From Bayesian neural networks to Dropout

- Place **prior** $p(W)$ dist. on weights, making these r.v.s
- Given dataset $X, Y$, the r.v. $W$ has a **posterior**: $p(W|X, Y)$
- Which is difficult to evaluate—many great researchers tried

- Can define **simple distribution** $q_M(\cdot)$ and approximate

$$q_M(W) \approx p(W|X, Y)$$
Some theory

From Bayesian neural networks to Dropout

- Place prior $p(W)$ dist. on weights, making these r.v.s
- Given dataset $X, Y$, the r.v. $W$ has a posterior: $p(W|X, Y)$
- Which is difficult to evaluate—many great researchers tried
- Can define simple distribution $q_M(\cdot)$ and approximate
  \[ q_M(W) \approx p(W|X, Y) \]
Some theory

From Bayesian neural networks to Dropout

- Place **prior** $p(W)$ dist. on weights, making these r.v.s
- Given dataset $X, Y$, the r.v. $W$ has a **posterior**: $p(W|X, Y)$
- Which is difficult to evaluate—many great researchers tried
- Can define **simple distribution** $q_M(\cdot)$ and approximate
  $$q_M(W) \approx p(W|X, Y)$$
From Bayesian neural networks to Dropout

- Place **prior** $p(W)$ dist. on weights, making these r.v.s
- Given dataset $X, Y$, the r.v. $W$ has a **posterior**: $p(W|X, Y)$
- Which is difficult to evaluate—many great researchers tried
- Can define **simple distribution** $q_M(\cdot)$ and approximate
  $$q_M(W) \approx p(W|X, Y)$$
- This is called **approximate variational inference.**
Theorem (Dropout as approximate variational inference)

Define \( q_M(W) := M \cdot \text{diag}(\text{Bernoulli}) \)

with variational parameter \( M \).

The optimisation objective of (stochastic) variational inference with \( q_M(W) \) is identical to the objective of a dropout neural network.

Proof.

See Gal [2016].
Theorem (Dropout as approximate variational inference)

Define

\[ q_M(W) := M \cdot \text{diag}(\text{Bernoulli}) \]

with variational parameter \( M \).

The optimisation objective of (stochastic) variational inference with \( q_M(W) \) is identical to the objective of a dropout neural network.

Proof.

See Gal [2016].

Implementing **inference** with \( q_M(W) \)

\[ = \]

Implementing **dropout training**.

Line to line.
Some theory

Theorem (Dropout as approximate variational inference)

Define
\[ q_M(W) := M \cdot \text{diag}(\text{Bernoulli}) \]

with variational parameter \( M \).

The optimisation objective of (stochastic) variational inference with \( q_M(W) \) is identical to the objective of a dropout neural network.

Corollary (Model uncertainty with dropout)

Given \( p(y^*|f^W(x^*)) = \mathcal{N}(y^*; f^W(x^*), \tau^{-1}I) \) for some \( \tau > 0 \), the model’s predictive variance can be estimated with the unbiased estimator:

\[
\tilde{\text{Var}}[y^*] := \tau^{-1}I + \frac{1}{T} \sum_{t=1}^{T} f^\hat{W}_t(x^*)^T f^\hat{W}_t(x^*) - \tilde{E}[y^*]^T \tilde{E}[y^*]
\]

with \( \hat{W}_t \sim q^*_M(W) \).
In practical terms\(^1\), given point \(x\):

- drop units at test time
- repeat 10 times
- and look at mean and sample variance.
- Or in Python:

```python
y = []
for _ in xrange(10):
    y.append(model.output(x, dropout=True))
y_mean = numpy.mean(y)
y_var = numpy.var(y)
```

\(^1\)Friendly introduction given in [yarin.co/blog](http://yarin.co/blog)
Example uncertainty in deep learning

What would be the CO$_2$ concentration level in Mauna Loa, Hawaii, in 20 years’ time?

- Normal dropout:

- Same network, Bayesian perspective:
Example uncertainty in deep learning

What would be the CO$_2$ concentration level in Mauna Loa, Hawaii, in 20 years’ time?

- Normal dropout:

- Same network, Bayesian perspective:
Example uncertainty in deep learning

What would be the CO₂ concentration level in Mauna Loa, Hawaii, in 20 years’ time?

Normal dropout: ▶ Bayesian perspective:

What can we do with this?

▶ Interpretability & AI safety
▶ Principled deep learning extensions
▶ Deep learning in small data domains
What would be the CO$_2$ concentration level in Mauna Loa, Hawaii, in 20 years’ time?

What can we do with this?

- Interpretability & AI safety
- Principled deep learning extensions
- Deep learning in small data domains
  - Cancer diagnosis
Active learning of images [Gal, Islam & Ghahramani, 2017]
E.g. diagnose melanoma with a handful of images.
Choose $x^*$ that maximises acquisition functions $a(x)$:

$$x^* = \arg\max_{x \in D_{pool}} a(x)$$

E.g. points that maximise uncertainty. But, *which uncertainty?*

- *Aleatoric uncertainty* captures noise inherent in the data
- *Epistemic uncertainty* captures model’s lack of knowledge
- *Predictive uncertainty* captures the sum of the two

Figures adapted from Hanna M. Wallach (Cambridge, UMassAmherst)
Acquisition functions for classification

Choose $x^*$ that maximises acquisition functions $a(x)$:

$$x^* = \arg\max_{x \in D_{\text{pool}}} a(x)$$

Possible measures of uncertainty in classification:

- **Predictive entropy ($\mathbb{H}[y|x, D_{\text{train}}]$)**
  
  $$a_{\text{PE}}(x) = -\sum_c p(y = c|x, D_{\text{train}}) \log p(y = c|x, D_{\text{train}})$$

- **Information gained about the model parameters ($\mathbb{H}[y, W|x, D_{\text{train}}]$)**
  
  $$a_{\text{MI}}(x) = \mathbb{H}[y|x, D_{\text{train}}] - \mathbb{E}_{p(W|D_{\text{train}})} [\mathbb{H}[y|x, W]]$$

- **Variation ratios**
  
  $$a_{\text{VR}}(x) = 1 - \max_y p(y|x, D_{\text{train}})$$

- **Random acquisition (baseline):** $a_{\text{U}}(x) = \text{unif}()$
Want to classify dogs vs. cats given image $\mathbf{x}$ with models $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3$

- Stochastic forward passes give **probability vectors** for each model:
  1. $(1, 0), \ldots, (1, 0)$
  2. $(0.5, 0.5), \ldots, (0.5, 0.5)$, and
  3. $(1, 0), (0, 1), (1, 0), \ldots, (0, 1)$
Acquisition functions intuition

Want to classify dogs vs. cats given image $\mathbf{x}$ with models $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3$

- Stochastic forward passes give **probability vectors** for each model:
  1. $(1, 0), \ldots, (1, 0)$
  2. $(0.5, 0.5), \ldots, (0.5, 0.5)$, and
  3. $(1, 0), (0, 1), (1, 0), \ldots, (0, 1)$

What’s the epistemic uncertainty for each model? What’s the predictive uncertainty for each model?
Acquisition functions intuition

Want to classify dogs vs. cats given image $\mathbf{x}$ with models $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3$

- Stochastic forward passes give **probability vectors** for each model:
  1. $(1, 0), \ldots, (1, 0)$
  2. $(0.5, 0.5), \ldots, (0.5, 0.5)$, and
  3. $(1, 0), (0, 1), (1, 0), \ldots, (0, 1)$

**What’s the epistemic uncertainty?** models $\mathcal{M}_1$ and $\mathcal{M}_2$ are confident about the output. Model $\mathcal{M}_3$ is uncertain.

**What’s the predictive uncertainty?** $\mathcal{M}_1$ has low uncertainty, $\mathcal{M}_2$ and $\mathcal{M}_3$ have high uncertainty.
Want to classify dogs vs. cats given image $\mathbf{x}$ with models $M_1, M_2, M_3$

- Stochastic forward passes give probability vectors for each model:
  1. $(1, 0), \ldots, (1, 0)$
  2. $(0.5, 0.5), \ldots, (0.5, 0.5)$, and
  3. $(1, 0), (0, 1), (1, 0), \ldots, (0, 1)$

**What’s the epistemic uncertainty?** models $M_1$ and $M_2$ are confident about the output. Model $M_3$ is uncertain.

**What’s the predictive uncertainty?** $M_1$ has low uncertainty, $M_2$ and $M_3$ have high uncertainty.

**Acquisition functions intuition:**

- $M_1$: all acquisition functions give low uncertainty
- $M_2$: variation ratios and predictive entropy give high uncertainty; mutual information gives low uncertainty.
- $M_3$: all acquisition functions give high uncertainty
MNIST experiments

Test accuracy as a function of number of acquired images (up to 1K):

- BALD
- Var Ratios
- Max Entropy

using both a **Bayesian CNN (red)** and a **deterministic CNN (blue)**

Number of acquired images **to get to model error of %**: 

<table>
<thead>
<tr>
<th>% error</th>
<th>BALD</th>
<th>Var Ratios</th>
<th>Max Ent</th>
<th>Random</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>145</td>
<td>120</td>
<td>165</td>
<td>255</td>
</tr>
<tr>
<td>5%</td>
<td>335</td>
<td>295</td>
<td>355</td>
<td>835</td>
</tr>
</tbody>
</table>
Test error on MNIST with 1000 labelled training samples, for active learning (using simple LeNet) vs. **semi-supervised techniques**:

<table>
<thead>
<tr>
<th>Technique</th>
<th>Test error</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Semi-supervised:</strong></td>
<td></td>
</tr>
<tr>
<td>SS Embedding (Weston et al., 2012)</td>
<td>5.73%</td>
</tr>
<tr>
<td>DGN (Kingma et al., 2014)</td>
<td>2.40%</td>
</tr>
<tr>
<td>Γ Ladder Network (Rasmus et al., 2015)</td>
<td>1.53%</td>
</tr>
<tr>
<td>Virtual Adversarial (Miyato et al., 2015)</td>
<td>1.32%</td>
</tr>
<tr>
<td><strong>Active learning with various acquisitions:</strong></td>
<td></td>
</tr>
<tr>
<td>Random</td>
<td>4.66%</td>
</tr>
<tr>
<td>BALD</td>
<td>1.80%</td>
</tr>
<tr>
<td>Max Entropy</td>
<td>1.74%</td>
</tr>
<tr>
<td>Var Ratios</td>
<td>1.64%</td>
</tr>
</tbody>
</table>
Active learning of images [Gal, Islam & Ghahramani, 2017]

E.g. diagnose melanoma with a handful of images:

Performance vs. acquisition
Active learning of images [Gal, Islam & Ghahramani, 2017]
E.g. diagnose melanoma with a handful of images:

# acquired positive examples vs. acquisition
Most exciting is work to come:

- What is *interesting* data to *label*? (when model is uncertain)
- Active learning in real-world *medical applications*

*and much, much, more.*
Most exciting is work to come:

- What is interesting data to label? (when model is uncertain)
- Active learning in real-world medical applications

and much, much, more.

Thank you for listening.
References


▶ Y Gal, R McAllister, C Rasmussen, “Improving PILCO with Bayesian Neural Network Dynamics Models”, DEML workshop, ICML (2016)


▶ Y Li, Y Gal, “Dropout Inference in Bayesian Neural Networks with Alpha-divergences”, ICML (2017)


▶ A Shah, Y Gal, “Invertible Transformations for Bayesian Neural Network Inference” (2017)

▶ and more…