Uncertainty in Bayesian Neural Networks

Javier Antorán, Xiping Liu, Efstratios Markou, Xianru Zheng; {ja666, xl445, em626, xz396}@cam.ac.uk

Why be Bayesian?

- Weight uncertainty: knowing what we don’t know.
- Balance modelling capacity and simplicity.

Figure: A NN trained with MAP inference presents low predictive entropy when misclassifying ood samples.

Approximate Inference Methods

The posterior over $w$ is intractable for neural nets. We consider the following approximations.

- Bayes by Backprop [1]

  \[
  \text{ELBO} \approx L_{\text{BBP}}(\mu, \sigma) = \frac{1}{N} \sum_{i=1}^{N} \left[ \log p(y|x, w^{(i)}) - \log q(w^{(i)}|\mu, \sigma) + \log p(w^{(i)}) \right]
  \]
  
  $w^{(i)} = \mu + \sigma \cdot e^{(i)}$, $e^{(i)} \sim N(0, I)$

- MC Dropout [3]

  \[
  \text{ELBO} \approx L_{\text{dropout}}(m) = \log p(y|x, w) - \lambda \|m\|_2^2
  \]
  
  $w = m \odot z$, $z \sim \text{Bernoulli}(p_{\text{drop}})$

- Stochastic Gradient Langevin Dynamics [4]

  \[
  \Delta w^{(i)} = \frac{\epsilon^{(i)}}{M} \nabla \log p(y|x, w^{(i)}) + \frac{\eta^{(i)}}{N_{\text{batch}} \epsilon^{(i)}} \sum_{n=1}^N \nabla \log p(y_n|x_n, w^{(i)}) + \eta^{(i)}
  \]
  
  \[
  \eta^{(i)} \sim N(0, \epsilon^{(i)} M)
  \]

Uncertainty Decomposition

Uncertainty caused by noise, or Aleatoric uncertainty, can be quantified as $E[\epsilon] \sigma_{\text{pred}}^2$ or $\mathcal{H}_a = \mathcal{E}[\epsilon|w]\mathcal{H}(y'|x'|w)$. Model or Epistemic uncertainty can be measured as $\mathcal{V}ar_{\epsilon|w}(\mu_{\text{pred}})$ or $\mathcal{H}_e = \mathcal{H}(y'|x') - \mathcal{H}_a [2]$.

Figure: Toy homoscedastic regression task. Data is generated by a GP with a RBF kernel ($\ell = 1, \sigma_f = 0.3$). We use a single-output FC network with one hidden layer of 200 ReLU units to predict the regression mean $\mu(x)$. A fixed log-$\sigma$ is learnt separately.

Figure: Toy heteroscedastic regression task. Data is generated by a GP with a RBF kernel ($\ell = 1, \sigma_f = 0.3 \cdot |x| + 2$). We use a two-head network with 200 ReLU units to predict the regression mean $\mu(x)$ and log-standard deviation log-$\sigma(x)$.

MNIST Classification

<table>
<thead>
<tr>
<th>Model</th>
<th>MAP</th>
<th>MAP Ensemble</th>
<th>BBP Gaussian</th>
<th>BBP Laplace</th>
<th>BBP Local</th>
<th>MC Dropout</th>
<th>SGLD</th>
<th>P-SGLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log Likelihood</td>
<td>-572.90</td>
<td>-496.54</td>
<td>-1100.29 -1008.28 -892.85 -1086.43</td>
<td>-435.458 -828.29 -661.25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error %</td>
<td>1.58</td>
<td>1.53</td>
<td>2.60</td>
<td>2.38</td>
<td>2.28</td>
<td>2.61</td>
<td>1.37</td>
<td>1.76</td>
</tr>
</tbody>
</table>

Table: MNIST test results for methods under consideration. We approximate $E_{\text{pred}}[\bar{y}|y', x']$ with 100 MC samples. We use a FC network with two 1200 unit ReLU layers. If unspecified, the prior is Gaussian. P-SGLD uses RMSprop preconditioning.

Figure: Comparison of each method’s decomposed predictive entropy on ood samples: rotated MNIST digits. t is the correct class.

References