Topics in Computer Vision and Deep Learning

A chronological overview of influential publications

Uncertainty Estimation and Probabilistic Neural Networks

Mats Steinweg
KTH Royal Institute of Technology
matsstei@kth.se

September 16, 2019

Table of Contents

Weight Uncertainty in Neural Networks

ICML 2015

Summary

In this paper, a new algorithm for training probabilistic neural networks, referred to as *Bayes by Backprop*, is proposed. The algorithm improves upon the work proposed by Graves in 2011 (see paper "Practical Variational Inference in Neural Networks) by computing unbiased gradients and allowing for non-gaussian priors. Training a network using Bayes by Backprop corresponds to an ensemble of networks with weights drawn from a shared posterior distribution. Evaluated on MNIST, the algorithm performs on-par with networks regularized using dropout.

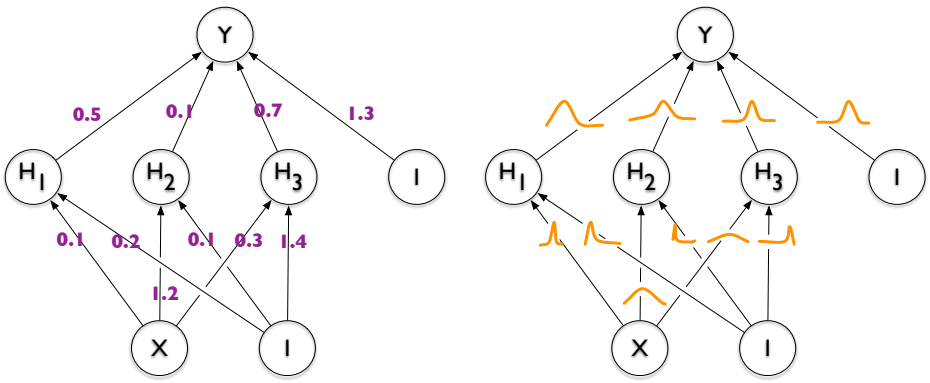

Figure 1: Comparison of network trained using conventional backpropagation and a network trained using Bayes by Backprop. Instead of learning a point estimate of each weight, the full posterior distribution of each weight given the training data is learned.

Main Contributions

- **Unbiased Monte Carlo Gradients** The authors put forward a simple formula for the gradients of the variational free energy w.r.t the parameters of the variational distribution that reuses the gradients needed for conventional backpropagation.
- **Scale Mixture Prior** Contrary to other works that tried to optimize the parameters of the prior distribution of the weights, a mixture of two gaussian is proposed that alleviates the need for optimization of more parameters.
- **Mini-Batch Re-Weighting** Based on the assumption that uniform sampling of mini-batches can result in a non-uniform distribution of the cost function, a weighting scheme is proposed that puts more emphasis on the prior in the beginning of the training. The more data is observed, the more influence is then assigned to the likelihood of the data.
- **Model Pruning** It is shown that the weights obtained using Bayes by Backprop display a significantly higher level of scarcity compared to conventional SGD. Consequently, a large number of weights can be removed from the network after training without impairing performance.

Implementation Details

- **Approximation of the Variational Free Energy** Following the popular approach of approximating the intractable posterior over the network's weights $P(\mathbf{w}|\mathcal{D})$ by a simpler variational distribution $q(\mathbf{w}|\theta)$, the cost function to be minimized corresponds to the variational free energy:

$$\begin{aligned}\mathcal{F}(\mathbf{w}|\theta) = &\quad \mathbf{KL}\left[q(\mathbf{w}|\theta) \,||\, P(\mathbf{w})\right] - \mathbb{E}_{q(\mathbf{w}|\theta)}\left[\log P(\mathcal{D}|\mathbf{w})\right] \\ \approx &\sum_{i=1}^{n} \log q(\mathbf{w}^{(i)}|\theta) - \log P(\mathbf{w}^{(i)}) - \log P(\mathcal{D}|\mathbf{w}^{(i)})\end{aligned}$$

  This approximation ensures that all terms in the cost function depend on the particular Monte Carlo sample to the weights $\mathbf{w}^{(i)}$.

- **Parameter Update** Given a variational gaussian posterior distribution, a sample of the weights is given by $\mathbf{w} = \mu + \log(1 + \exp(\rho)) \cdot \epsilon$ for $\epsilon \sim \mathcal{N}(0, I)$. The gradients w.r.t. the variational parameters $\theta = (\mu, \rho)$ are given by:

$$\begin{aligned}\Delta_\mu &= \frac{\partial f(\mathbf{w}, \theta)}{\partial \mathbf{w}} + \frac{\partial f(\mathbf{w}, \theta)}{\partial \mu} \\ \Delta_\rho &= \frac{\partial f(\mathbf{w}, \theta)}{\partial \mathbf{w}} \frac{\epsilon}{1 + \exp(-\rho)} + \frac{\partial f(\mathbf{w}, \theta)}{\partial \rho}\end{aligned}$$

Evaluation

- **MNIST Classification** Bayes by Backprop is used to train a 2-Layer ReLU CNN with 1200 units per layer and the performance is compared to the same architecture trained using conventional SGD and dropout SGD. The Bayes by Backprop variant with a gaussian mixture prior achieves a test error of 1.32% on-par with dropout SGD (1.36%).

- **Weight Scarcity & Model Pruning** It can be observed that the weight values obtained through Bayes by Backprop display a much wider spread compared to SGD and dropout. This phenomenon is illustrated in Figure 2. Consequently, a large fraction of the weights (up to 75%) can be removed based on the Signal-to-Noise ratio ($|\mu_i|/\sigma_i$) without negatively affecting the test error.

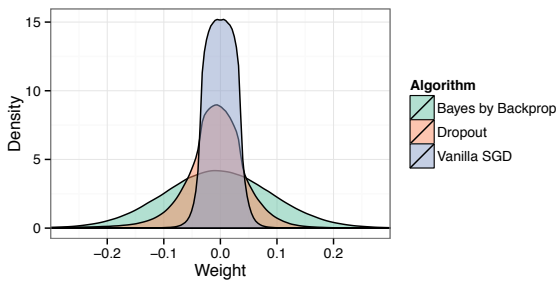

Figure 2: Histogram of the trained networks' weights.

References

This summary is solely based on my understanding of the original paper. All images used here are taken from the original paper as well. The paper can be found under the following link:
`https://arxiv.org/pdf/1505.05424.pdf`

Dropout as a Bayesian Approximation

ICML 2016

Summary

In this paper, the authors propose a new technique called *Monte Carlo Dropout* that re-uses information contained in conventional dropout neural networks to approximate predictive uncertainty. The algorithm is based on a link between gaussian processes and dropout networks and allows for the extraction of uncertainty information without an increase in computational complexity or a decrease in classification accuracy. Monte Carlo Dropout is shown to provide reliable uncertainty estimates on regression, classification and reinforcement learning problems.

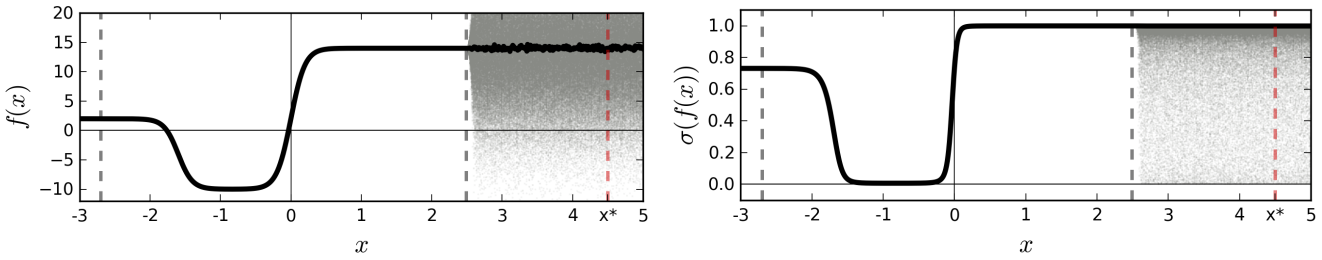

Figure 3: Illustration of input and output of a softmax function. Training data was used between the dashed gray lines. We can observe that the output of the softmax function on out-of-distribution data is much smoother than the input. A point x^* is consequently classified as class 1 with probability 1.0 despite a very noisy input function around that point. Uncertainty estimates could indicate the unfamiliarity of the network with points around x^*.

Main Contributions

- **Dropout and Gaussian Processes** The authors put forward a mathematical derivation of the link between dropout and gaussian processes that allows for a probabilistic interpretation of dropout. The paper shows that the objective of dropout corresponds to minimizing the KL divergence between a variational distribution and the posterior of a gaussian process.
- **Monte Carlo Dropout** A method is put forward that allows for the extraction of uncertainty information from standard dropout networks by applying dropout not only during training but also during inference. Thus, uncertainty information can be obtained without any additional computational cost commonly associated with probabilistic neural networks.
- **Classification Performance** The performance of MC Dropout is evaluated on MNIST, showing how the obtained uncertainty measures create robustness in the classification of rotated digits and allow for more interpretable results.

Implementation Details

- **Variational Inference** In order to approximate the intractable posterior over the network's weights, a variational distribution is selected that resembles the dropout approach by randomly disabling weights according to binary values sampled from a Bernoulli distribution:

$$\mathbf{W}_i = \mathbf{M}_i \cdot \text{diag}([\mathbf{z}_{i,j}]_{j=1}^{K_i})$$
$$\mathbf{z}_{i,j} \sim \text{Bernoulli}(p_i) \text{ for } i = 1, ..., L,\ j = 1, ..., K_{i-1}$$

- **Model Uncertainty** As mentioned before, the model uncertainty can be obtained from networks trained with dropout using standard point-estimates of the network parameters. The output is then defined by the mean and the variance computed over several forward passes using the same dropout scheme that was used during training. The authors mention that the predictive distribution $q(\mathbf{y}^*|\mathbf{x}^*)$ is expected to be of multi-modal nature due to the underlying assumptions of the variational approximation. Consequently, the characterization by mean and variance can only give an idea of the full distribution's characteristics.

Evaluation

- **MNIST Classification** The evaluation is conducted on a LeNet architecture trained on the full MNIST dataset. During training dropout is applied before the last fully-connected layer and the dropout percentage is set to 0.5. The network is presented with a continuously rotated version of the digit 1 and 100 forward passes are evaluated per instance. The results of the experiment are displayed in Figure 4. We can observe that for digit instance 5-8, the network is unable to produce a consistent prediction. In contrast to a single forward pass that might give a relatively large softmax output, computing the variance over the output of several forward passes will in this case indicate the high level of uncertainty in the network's prediction.

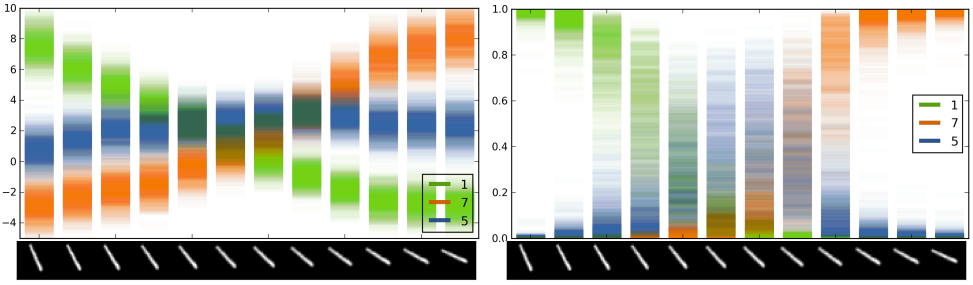

Figure 4: Softmax Input and Output for 100 forward passes of a rotated MNIST digit. The mean and variance of the Softmax output correspond to the prediction with uncertainty estimate obtained by MC Dropout.

References

This summary is solely based on my understanding of the original paper. All images used here are taken from the original paper as well. The paper can be found under the following link:
`https://arxiv.org/pdf/1506.02142.pdf`

On Calibration of Modern Neural Networks

ICML 2017

Summary

The authors of this paper address the issue of miscalibration in modern neural networks. Miscalibration in the realm of deep learning-based classification is defined as the gap between the output probability of the network and the likelihood of the prediction's correctness. In other words, does the output of the network indeed correspond to the probability of a correct prediction? The results presented indicate that popular modifications in modern deep learning architectures lead to increased miscalibration despite considerably higher classification scores. In order to counteract miscalibration, *Temperature Scaling*, a simple post-processing technique is proposed that improves the network calibration without influencing the classification score.

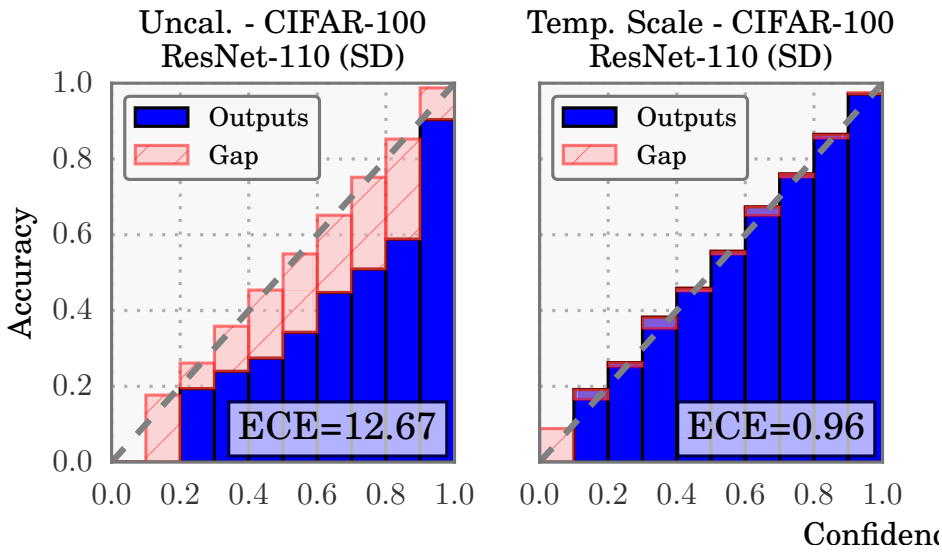

Figure 5: Reliability diagram of a ResNet-110 evaluated on CIFAR-100 with and without Temperature Scaling. The uncalibrated network outputs are misaligned with their likelihood of correctness which results in a large *Expected Calibration Error* (ECE). The proposed calibration method counteracts the misalignment and produces reliable confidence scores.

Main Contributions

- **Factors of Miscalibration** The authors present an investigation of the influence of several parameters such as model capacity, batch normalization and weight decay on the miscalibration of neural networks.
- **Method Evaluation** Different post-processing calibration methods are evaluated on a variety of image classification dataset.
- **Temperature Scaling** Temperature Scaling is proposed as a new approach to simply and efficiently calibrating neural networks.

Implementation Details

- **Reliability Diagrams** To illustrate the degree of miscalibration of the network, the expected prediction correctness is plotted against the network's confidence. To estimate the expected correctness from a finite number of samples, the confidence scores are assigned to M bins of size 1/M. A perfectly calibrated network corresponds to a diagonal reliability diagram. Deviations from the diagonal indicate over- or under-confident network predictions.
- **Expected Calibration Error** an approximation of ECE is applied to get a scalar measure indicating the degree of miscalibration. The ECE is defined as the expected difference between confidence and accuracy. This measure is approximated by taking the weighted average of each of the bins' confidence gap in the reliability diagram.
- **Temperature Scaling** The proposed calibration method is a simple extension of *Platt Scaling*. A single parameter T (referred to as the temperature) is used to scale the logits output of the network before feeding it to the softmax function. This parameter is optimized on the validation set using a cross entropy loss. The calibrated confidence score for a logits vector $\mathbf{z}_i$ is given by:

$$\hat{q}_i = \max_k \sigma_{\text{SM}}(\mathbf{z}_i/T)^{(k)}$$

Evaluation

- **Observing Miscalibration** The authors evaluate the influence of certain parameters on miscalibration by fixing network and training setup and observing the output for a single variable parameter.
  1. Model Capacity: Both increased width and depth induce higher ECEs. This might stem from the fact that a network can further lower a cross entropy loss by becoming more certain of its predictions.
  2. Batch Normalization: Despite positively influencing the accuracy, BN negatively affects the network's calibration. However, the authors provide no suggestions as to why that is the case.
  3. Weight Decay: While most modern neural networks are trained with little or no weight decay, a higher regularization parameter λ shows to positively affect the network's ECE.

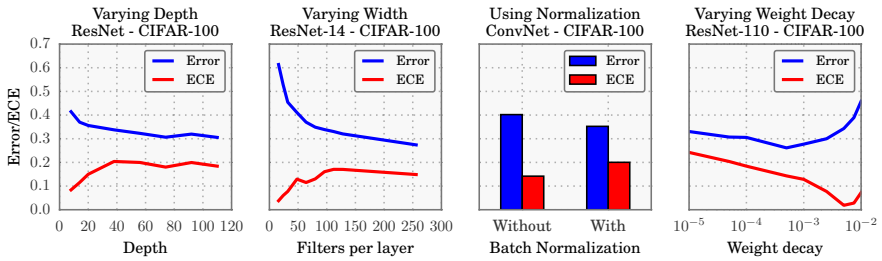

Figure 6: Classification Error and ECE of a ResNet-110 trained on CIFAR-100 as a function of different parameters affecting network and training process.

References

This summary is solely based on my understanding of the original paper. All images used here are taken from the original paper as well. The paper can be found under the following link:
`http://proceedings.mlr.press/v70/guo17a/guo17a.pdf`