On Bayesian Neural Networks at Finite Temperature

Robert J.N. Baldock[1, *]

[1] *Theory and Simulation of Materials (THEOS), and National Centre for Computational Design and Discovery of Novel Materials (MARVEL), École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland*

(Dated: January 2019)

ABSTRACT

In this blog post I examine the Bayesian formulation of neural networks due to MacKay [1]. I consider how the noise level (or "temperature") T in that formulation affects the posterior distribution. In particular, I highlight an approximate method for performing model selection on deep NNs.

I. STRUCTURE OF THIS BLOG POST

1. Outline of the Bayesian formulation of feed forward neural networks.
2. An analogy with physics, introducing an "energy" function for the network, and a "temperature", which controls the noise level.
3. Results section
4. Sampling methods (code in this repository). I built an HMC version of an accelerated sampling method from materials simulation, called Replica Exchange Molecular Dynamics which simultaneously explores the behaviour of a model across a wide range of temperatures.

II. BAYESIAN FORMULATION OF FEED FORWARD NEURAL NETWORKS AND TRAINING

In training a classifier one typically uses a minimisation algorithm to find a local minimum of a cost function defined over the parameters of the neural network model. If we use a "softmax" then the activation of the output units sum to one, and can be interpreted as the Bayesian probability, assigned by the network, to the assertion that the input belongs to each individual class. We can write this as $\text{prob}\,(q|\mathbf{x},\mathbf{w})$ where q is the index of the class, $\mathbf{x}$ is the single data example to be classified and $\mathbf{w}$ represents the parameters of the neural network. For data in the training set $D = \{\mathbf{x}_i, t_i\}$ where $\mathbf{x}_i$ is an input data example and t_i the corresponding class label, then the probability that the network classifies $\mathbf{x}_i$ correctly is $\text{prob}\,(q = t_i|\mathbf{x}_i,\mathbf{w})$. This probability is a function of the weights $\mathbf{w}$. The probability that the network classifies the complete data set correctly is called the *likelihood* of the data and is given by $\prod_i \text{prob}\,(q = t_i|\mathbf{x}_i,\mathbf{w})$.

If we assign a Gaussian prior with standard deviation σ to the parameters $\mathbf{w}$ then the *posterior* probability of the data is given by

$$\text{prob}\,(\mathbf{w}|D) \propto \prod_i \text{prob}\,(q = t_i|\mathbf{x}_i,\mathbf{w}) e^{-\frac{\mathbf{w}^2}{2\sigma^2}}. \qquad (1)$$

The posterior probability of the weights are therefore maximised when we minimise

$$J(\mathbf{w}|D) = -\sum_i \log\left[\text{prob}\,(q = t_i|\mathbf{x}_i,\mathbf{w})\right] + \frac{\mathbf{w}^2}{2\sigma^2} \qquad (2)$$

which can immediately be recognised as the cross entropy loss function with L2 regularisation. The minimum of (2) is called the maximum a posteriori solution.

In this blog post I will be using a uniform prior for $\mathbf{w}$

$$\text{prob}\,(\mathbf{w}) = \begin{cases} \frac{1}{\prod_{i=1}^{d}\sigma_i}, & |w_i| < \sigma_i/2\,\forall\, i, \\ 0, & \text{Elsewhere.} \end{cases} \qquad (3)$$

Within the bounds of this prior cost function is given by

$$J(\mathbf{w}|D) = -\sum_i \left(\log\left[\text{prob}\,(q = t_i|\mathbf{x}_i,\mathbf{w})\right] - \log\sigma_i\right). \qquad (4)$$

Outside the bounds of the prior the cost function is infinite. Minimising the cost function will simply mean minimising the likelihood within these bounds.

III. INTRODUCING ENERGY AND TEMPERATURE

One may define the "potential energy" over the parameters of the network [1]

$$E(\mathbf{w}|D) = -\sum_i \log\left[\text{prob}\,(q = t_i|\mathbf{x}_i,\mathbf{w})\right] \qquad (5)$$

and a "thermal" posterior distribution

$$\text{prob}\,(\mathbf{w}|T, D) \propto e^{-\frac{1}{T}E(\mathbf{w}|D)}\,\text{prob}(\mathbf{w}). \qquad (6)$$

where we assign T controls the noise in our posterior. We name T the "temperature" in analogy with thermodynamic temperature.

[*] rjnbaldock@gmail.com

For this blog post, it is important to imagine how the thermal posterior behaves as we vary T. The distribution (6) tends towards the prior distribution for $T \to \infty$, and is equal to the true posterior distribution (1) for $T = 1$. At $T < 1$ (low temperature) the thermal posterior (6) is concentrated around the maximum a posteriori solution (the minimum of (2)).

A. Temperature vs batch size

It has recently been discovered that mini-batch learning improves generalisation. There has been a lot of interest in how the batch size in mini-batch learning controls the noise level during learning, and under what conditions the optimiser can be said to be approximately sampling from the posterior. Here though I wanted to explore instead using the temperature to control the noise in full batch training and as an alternative to early stopping.

IV. RESULTS

A. Hessian free Bayesian model selection for deep (or shallow) neural networks

I'll start with a quick overview of Bayesian model selection. Then I'll introduce the Laplace approximation (approximating a distribution by a Gaussian fitted to a given maximum of the true distribution). Finally, I'll show how you can compute the Bayesian evidence for that Gaussian distribution, by finding the determinant of the Hessian without calculating the Hessian directly.

1. Bayesian Model Selection

Imagine we have two different machine learning models M_1 and M_2, and a data set $D = \{\mathbf{x}, t_i\}$, or in vector notation $D = (\mathbf{x}, \mathbf{t})$ where $\mathbf{x}$ is a matrix of all the input data and $\mathbf{t}$ a vector of the corresponding labels.

In Bayesian statistics we are able to calculate the probability of each model given the data, $\text{prob}\left(M|\mathbf{x},\mathbf{t}\right)$. In Bayesian model selection you choose the model with the highest probability. Actually you never choose, you believe them all do different extents, according to their probabilities.

The ratio of the probabilities for M_1 and M_2 can be found as follows

$$\text{prob}\left(M,\mathbf{t}|\mathbf{x}\right) = \text{prob}\left(M|\mathbf{x},\mathbf{t}\right)\text{prob}\left(\mathbf{t}|\mathbf{x}\right) \tag{7}$$

$$= \text{prob}\left(\mathbf{t}|\mathbf{x},M\right)\text{prob}\left(M|\mathbf{x}\right) \tag{8}$$

$$\Rightarrow \frac{\text{prob}\left(M_1|\mathbf{x},\mathbf{t}\right)}{\text{prob}\left(M_2|\mathbf{x},\mathbf{t}\right)}\frac{\text{prob}\left(\mathbf{t}|\mathbf{x}\right)}{\text{prob}\left(\mathbf{t}|\mathbf{x}\right)} = \frac{\text{prob}\left(\mathbf{t}|\mathbf{x},M_1\right)}{\text{prob}\left(\mathbf{t}|\mathbf{x},M_2\right)}\frac{\text{prob}\left(M_1\right)}{\text{prob}\left(M_2\right)} \tag{9}$$

$$\Rightarrow \frac{\text{prob}\left(M_1|\mathbf{x},\mathbf{t}\right)}{\text{prob}\left(M_2|\mathbf{x},\mathbf{t}\right)} = \frac{\text{prob}\left(\mathbf{t}|\mathbf{x},M_1\right)}{\text{prob}\left(\mathbf{t}|\mathbf{x},M_2\right)}\frac{\text{prob}\left(M_1\right)}{\text{prob}\left(M_2\right)} \tag{10}$$

The last term on the right in (10) is the ratio of our priors for the models. Typically we might initially have no preference between M_1 and M_2, in which case this last term is equal to 1.

In theory one can calculate the "marginal likelihood" of the model, $\text{prob}\left(\mathbf{t}|\mathbf{x},M\right)$, as follows

$$\text{prob}\left(\mathbf{t}|\mathbf{x},M\right) = \int d\mathbf{w}\,\text{prob}\left(\mathbf{t},\mathbf{w}|\mathbf{x},M\right) \tag{11}$$

$$= \int d\mathbf{w}\,\text{prob}\left(\mathbf{t}|\mathbf{w},\mathbf{x},M\right)\text{prob}\left(\mathbf{w}\right) \tag{12}$$

$$= \int d\mathbf{w}\,e^{-J(\mathbf{w}|D)} \tag{13}$$

In (13) we have made the connection with (4). There are lots of clever ways to calculate this integral. For deep neural networks it is almost always highly multimodal.

2. The Laplace approximation

One way to compute integral (12) is to make the Laplace approximation. We move to a local (ideally global) maximum of the posterior $\text{prob}\left(\mathbf{t}|\mathbf{w},\mathbf{x},M\right)\text{prob}\left(\mathbf{w}|M\right)$, which is at $\mathbf{w}_{\text{MP}}$. We approximate the integral (12) as the height of the peak of the integrand posterior, times its parameter space volume $\prod_{i=1}^{d}\sigma_{\text{MP},i}$ where $\{\sigma_{\text{MP},i}^{-2}\}$ are the eigenvalues of the Hessian of $J\left(\mathbf{w}_{\text{MP}}|D\right)$ calculated at $\mathbf{w}_{\text{MP}}$. In this case we can approximate the integral (12) as

$$\text{prob}\left(\mathbf{t}|\mathbf{x},M\right) \simeq e^{-J(\mathbf{w}_{\text{MP}}|D)}\prod_{i=1}^{d}\sigma_{\text{MP},i} \tag{14}$$

The value of $J\left(\mathbf{w}_{\text{MP}}|D\right)$ is directly obtained at the conclusion of minimisation. All we need is the $\prod_{i=1}^{d}\sigma_{\text{MP},i}$ but we don't want to calculate or diagonalise the Hessian. The cost of doing so is cubic in the number of parameters, which could easily reach the millions for a production model.

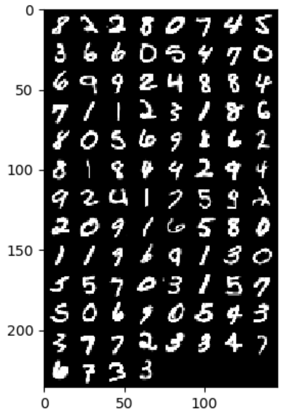

FIG. 1. 100 random data samples from the MNIST training set. These images have been normalised and rescaled down from 28x28 to 16x16.

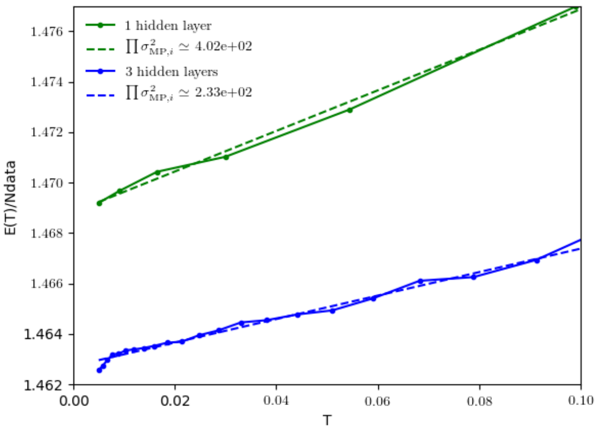

FIG. 2. Hessian free estimation of the determinant of the Hessian of the cost function. There are 5000 data points, so the y axis has been scaled by a factor of 1/5000, and the difference in the energy is in fact 33. Interestingly the Hessians of the cost functions have almost equal determinant.

3. *Hessian free estimation of the determinant of the Hessian*

Since we have assumed that the cost function is quadratic around $\mathbf{w}_{\mathrm{MP}}$ we can make use of a result from statistical mechanics, to calculate $\prod_{i=1}^{d} \sigma_{\mathrm{MP},i}$. Assuming the quadratic approximation does to the cost function does not place a lot of weight outside the bounds of the uniform prior (reasonable since there are many minima within that space) then we obtain

$$\prod_{i=1}^{d} \sigma^2_{\mathrm{MP},i} \simeq \frac{d\langle E \rangle}{dT} \tag{15}$$

where the average is taken over the thermal posterior (6). This is almost an equality in the limit of small T. I need to check but I think that for a Gaussian prior I can obtain a similar result $\prod_{i=1}^{d} \sigma^2_{\mathrm{MP},i} \simeq \frac{d\langle J \rangle}{dT}$ if I also raise the prior to the power $1/T$ in (6).

I applied a new formulation of Replica Exchange Molecular Dynamics with Hamiltonian Monte Carlo which I developed (see Section V), to two NN classifiers. My approach allowed me to extract curves for $E(T)$. These networks and the MNIST data have already been described in Section **??**.

For computational speed the MNIST data images have been rescaled down from 28 x 28 to 16 x 16 (256 input dimensions), and normalisation has been applied. A random selection of 100 rescaled images is shown in Figure 1.

I had 500 data points for each character type (5000 in total). I used 40 logistic neurons per hidden layer, with a softmax on the final layer, and investigated networks with one and three hidden layers. Following the standard procedure I initialised each weight and bias uniformly at random inside the region $[-\frac{1}{\sqrt{k}}, \frac{1}{\sqrt{k}}]$ where k is the fan in of the neuron to which the weight points. The fan in includes the bias. The prior space was of the same shape but 1000 times wider. This was chosen because full batch optimisation of the cost function made some weights nearly this large. I believe three layers counts as a deep network. The results are shown in Figure 2.

Taking the minimum Energy values from these same data sets I obtain

$$\frac{\text{prob (1 H Layer)}}{\text{prob (3 H Layers)}} = \frac{e^{-E_1}}{e^{-E_2}} \frac{\prod_{i=1}^{d} \sigma^{(3\,layers)}_{prior,\,i}}{\prod_{i=1}^{d} \sigma^{(1\,layer)}_{prior,\,i}} \frac{\prod_{i=1}^{d} \sigma^{(1\,layer)}_{\mathrm{MP},i}}{\prod_{i=1}^{d} \sigma^{(3\,layers)}_{\mathrm{MP},i}} \tag{16}$$

$$\simeq \frac{e^{-7346}}{e^{-7313}} \frac{e^{2435060}}{e^{1410560}} \frac{\sqrt{402}}{\sqrt{233}} \tag{17}$$

$$\simeq 1 \times 10^{444920}. \tag{18}$$

We can see that the "Occam's razor" term $\frac{\prod_{i=1}^{d} \sigma_{\mathrm{MP},i}}{\prod_{i=1}^{d} \sigma_{prior,\,i}}$ showed overwhelming preference for the 1 hidden layer network, which did nearly as well on the classification task.

What might well make this a useful approach is that relative prior probabilities of models can differ so enormously. Finally the sensitivity to $\prod_{i=1}^{d} \sigma_{\mathrm{MP},i}$ is low: an answer within an order of magnitude may well be perfectly acceptable.

B. Discussion

V. SAMPLING METHODS

I developed a Hamiltonian Monte Carlo (HMC) formulation of a technique called Replica Exchange Molecular Dynamics (REMD) for use with NN models and study some classifiers for MNIST digits. REMD is a method for exploring the behaviour of a sampler simultaneously across a range of temperatures. Samplers periodically attempt to exchange their current points in parameter space with samplers at neighbouring temperatures. This accelerates parameter space sampling at low temperatures because each set of coordinates spends some of the time at high temperatures where they move around much more rapidly.

I'll add detail to this section *very soon.* For now, suffice it to say that I was inspired by Radford Neal's work on HMC in neural networks. I explored only the parameters of the model, keeping the hyper parameters fixed. You can read about HMC here http://www.mcmchandbook.net/HandbookChapter5.pdf and Replica Exchange Molecular Dynamics [2].

[1] D. J. MacKay, Neural computation **4**, 448 (1992).

[2] R. H. Swendsen and J.-S. Wang, Phys. Rev. Lett. **57**, 2607 (1986).