Dropout as a Bayesian Approximation: Clarification

Sourav Sahoo, Subhankar Chakraborty, Sumanth R Hegde

March 21, 2020

0.1 How are you going from dropout in NN to approximation of GP and thus explaining uncertainty? You have shown all the proofs but it would be great if you would have pointed out briefly the connection between dropout in NN and approximation in GP in your language after all the proofs.

The connection can be made with the following key points:

1. **Intention**: A neural network (NN) trained with dropout has the following characteristic: At each forward pass during training, we randomly set some input and hidden units to zero. The training proceeds by optimising a loss function which typically has some regularisation terms. The intention is to obtain a measure of true uncertainty in the model prediction. Towards this, we try to relate GP theory, where uncertainties are modelled naturally, to this setting, so that we can make the uncertainty estimates directly from standard NNs with dropout.

2. **The approximation**: The approximation of the GP involves the idea of variational inference. The task is now, as follows:

   2.1. Find a suitable approximate posterior q(w) and kernel function/ covariance function K.

   2.2. Parameterize the posterior.

   2.3. Convert GP to an optimization setting using variational inference.

3. The first idea is in approximating the kernel function (ref. page 2 of the paper) using Monte Carlo integration. *How does this relate to our NNs?* The associated feature map for the kernel can be easily found. (ref. Section 3.1 in appendix [1]). The feature mapping turns out to be exactly equal to the transformation by the hidden layer (upto scale). This agrees with intuition, that the hidden layer is a transformation performed on the input space and the inputs are classified in this transform space.

4. Now, we proceed to choose the approximate posterior $q(\mathbf{w})$. The choice made is a two-component GMM for the weights as explained in the next section. We now refer back to the objective: relating NN with dropout and L_2 regularisation to GPs.

5. **Reparameterisation** : We introduce additional Bernoulli random variables to reparameterise the objective. The objective, the log evidence lower bound is massaged into one resembling the dropout model. The reader can refer to eq. 14 in [1] for the governing equations. The following are key insights :

   5.1. The weights are expressed in terms of the variational parameters M_1, M_2 and in terms of standard normal random vectors using the Gaussian reparameterisation trick.

   5.2. The integrals involved in the objective are now replaced by Monte Carlo estimates, with only **one** sample. The reason for this choice, beyond the fact that it befits the proof, is unknown.

   5.3. Variance σ of the underlying Gaussians is set to nearly zero. This can be understood by simplifying the expression for weights (say W_1) in this limit (section 3.4 in [1]) :

$$W_1^n \approx diag(z_1^n) M_1$$

where the superscript indicates that we are dealing with random variable realisations. Thus, for each realisation of z_1, some rows of W_1 are dropped to zero, while others retain the value at this iteration (M_1 is a variational parameter over which the objective is optimised) . This is the exact setting of a dropout network. The same has been quoted in the paper as follows :

> *Note that even though our approximating distribution is, in effect, made of a sum of two point masses, each point mass with zero variance, the mixture does not have zero variance. It has the variance of a Bernoulli random variable, which is transformed through the network. This choice of approximating distribution results in the dropout model.*

6. **Uncertainty estimation**: *How do we obtain uncertainty estimates from dropout?*. The goal was after all to obtain the predictive distribution :

$$p(y^*|x^*, \mathbf{X}, \mathbf{Y}) \approx \int p(y^*|x^*, \mathbf{w}) q(\mathbf{w}) d\mathbf{w}$$

The integral can still be intractable even if q($\mathbf{w}$) is tractable, and thus we approximate this with Monte Carlo integration :

$$p(y^*|x^*, \mathbf{X}, \mathbf{Y}) \approx \frac{1}{T} \sum_{t=1}^{T} p(y^*|x^*, \mathbf{w}_t)$$

A closed form expression can be readily obtained and the reader can refer to eq. 22 in [1] for the expression for the log likelyhood. The first and the second moments can be obtained in a similar fashion using Monte Carlo integration. Thus, for obtaining the predictive, we perform T forward passes at test time and record the predictions.

0.2 Why $(q(W1), q(W2))$ etc are Gaussian mixture distribution with two-components? Why not multiple components? What will happen if you use multiple component GMM?

The paper aims to prove that a *neural network with dropout is essentially an approximation of a Gaussian Process* under special circumstances. In a regular dropout neural network, with a probability p the weight is preserved and with a probability $1 - p$, it is dropped, where p is the dropout parameter. So, we do an approximation as mentioned in Equation 1.

$$q(\mathbf{w_q}) = p\mathcal{N}(\mathbf{m}_q, \sigma^2 I_K) + (1 - p)\mathcal{N}(0, \sigma^2 I_K) \tag{1}$$

This Gaussian mixture represents a similar situation as that of a neural network with dropout. Ignoring the variance (noise), the weight of the component corresponding to non-zero mean is p and that of the component with zero mean is $1 - p$. The choice of the Gaussian mixture having two components only, is done to "mimic" a neural network with dropout. If we consider a multiple component Gaussian Mixture, then the neural network approximation will not be valid.

0.3 Is it always true that a heavy-tailed input will give rise to a heavy-tailed output?

To clear out any ambiguity, we state our interpretation of the question, for the task of classification:

> Given an input distribution which is heavy tailed. Let $\mathbf{x}$ be an input belonging to a well represented class and the prediction distribution for $\mathbf{x}$ be p(y). Is p(y) heavy-tailed?

The softmax probability vector for an input simply depends on the extent to which various discriminatory features are manifested in the input. The predictive distribution p(y) is the distribution over *all possible softmax probability vectors*. To the best of our knowledge, this distribution need not follow any heavy tailed curve.

References

[1] Yarin Gal, Zoubin Ghahramani : *Dropout as a Bayesian Approximation: Appendix*, ICML, 2016.