Subnetwork Inference

Tony

April 2022

1 Introduction

In these notes I explain the paper Bayesian Deep Learning via Subnetwork Inference from Daxberger et al. (2021). I will also some notes on an idea that I have to apply SGMCMC only on a subnetwork chosen by the method proposed in the aforementioned paper.

2 Motivation

Overparameterization. It has been shown that near the local optima there can be different directions of the gradient providing the same performance (Maddox et al., 2020). NNs can also be extremely pruned without sacrificing accuracy (Frankle and Carbin, 2018; Panousis et al., 2019).

Subnetwork inference. There has already been some work that proved that inference can be applied effectively only in a small part of the network (Snoek et al., 2015; Izmailov et al., 2020), like over low-dimensional projections of the weights, or only the last layer of a NN.

3 Preliminaries

Let data $D = \{y, X\}$ and $p(w)$ the prior over the NN weights. Then the full posterior is the following:

$$p(w|D) = p(w|y, X) \approx p(y|X, w)p(w) \tag{1}$$

As we know the computation of the posterior is challenging and therefore we approximate it.

$$\begin{aligned} p(w|D) \approx & \, p(w_s|D) \prod_i \delta(w_r - \hat{w}_r) \\ \approx & q(w_s) \prod_r \delta(w_r - \hat{w}_r) \end{aligned} \tag{2}$$

The first equation in Eq. (2) breaks the posterior in a posterior only over a subnetwork of the full network $p(w_s|D)$ and some dirac functions $\delta(w_r - \hat{w_r})$. The dirac functions are over the remaining weights out of the subnetwork $w_s \in \mathbb{R}^s$ to keep them at fixed values $\hat{w}_r \in \mathbb{R}$. We further approximate $p(w_s)$ since it is still computationally challenging with an approximate distribution $q(w_s)$.

Note. The idea in Eq. (2) is to set the variance of the fixed w_rs to zero, in contrast with pruning methods that set the weights to zero. In that way we remove the variances of the weights but we keep their predictive mean to retain accuracy to the full model.

4 Linearized Laplace

This part of the notes is defined in more detail in my other notes. In this section I will briefly introduce linearized Laplace approximation. Let a neural network denoted by function $f : \mathbb{R}^I \to \mathbb{R}^O$. The weights of the NN have a prior that is fully factorised Gaussian $p(w) = N(w; 0, \lambda I)$. We find the MAP as follows:

$$\hat{w} = \arg\max_w [\log p(y|X, w) + \log p(w)] \tag{3}$$

The posterior can be approximated with a second order Taylor expansion around the MAP as follows:

$$\log p(w|D) \approx logp(\hat{w}|D) - \frac{1}{2}(w - \hat{w})^\intercal H(w - \hat{w}) \tag{4}$$

where H is the hessian matrix of the negative log-posterior density w.r.t. the network weights w:

$$H = N \cdot \mathbb{E}_{p(D)}[-\partial^2 \log p(y|X, w)/\partial w^2] + \lambda I \tag{5}$$

Then we can rewrite the approximate posterior as follows:

$$p(w|D) \approx q(w) = N(w; \hat{w}, H^{(-1)}) \tag{6}$$

then the Hessian can be replaced with the generalized Gauss-Newton matrix (GGN) $\tilde{H} \in \mathbb{R}^{D\times D}$ as follows:

$$\tilde{H} = \sum_{n=1}^{N} J_n^\intercal H_n J_n + \lambda I \tag{7}$$

where $J_n = \partial f(x_n, w)/\partial w \in \mathbb{R}^{O\times D}$ is the Jacobian of the model outputs $f(x_n, w) \in \mathbb{R}^O$ w.r.t. w, and $H_n = -\partial^2 \log p(y|f(x_n, w))/\partial^2 f(x_n, w) \in \mathbb{R}^{O\times O}$ is the Hessian of the negative log-likelihood w.r.t. model outputs. It has been proved that when we use a Gaussian likelihood, the Gaussian with a GGN precision matrix corresponds to the true posterior distribution when the NN is approximated with a first-order Taylor expansion around the MAP $\hat{w}$. Then the locally linearized function is:

$$f_{lin}(x, w) = f(x, \hat{w}) + \hat{J}(x)(w - \hat{w}) \tag{8}$$

where $\hat{J} = \partial f(x, \hat{w})/\partial \hat{w} \in \mathbb{R}^{O\times O}$ is the Jacobian. In this way we convert the BNN into a generalized linear model (GLM), where the Jacobian $\hat{J}$ acts as the basis function expansion.

5 Linearized Laplace Subnetwork Inference

In this section I present the procedure of inference in a subnetwork.

Step 1: Point Estimation. We train a NN and obtain a point estimate of the weights $\hat{w}$.

Step 2: Subnetwork Selection. We select a small subnetwork that ideally produces a predictive posterior closest to the full predictive posterior of the full network. The way that we do that is by minimizing the Wasserstein distance between the weight posterior of the subnetwork and the full network.

The goal is to minimize the discrepancy between the exact posterior of the full network and the subnetwork approximate posterior. The first challenge on that is that computing the exact posterior distribution remains intractable. The second challenge is that the KL divergence or the Hellinger distance is not very well defined for the Dirac delta distributions in Eq. (2). To overcome the first issue we resort to local linearization that we mentioned earlier, so we have the true posterior of the linearized model:

$$p(w|D) \simeq N(w; \hat{w}, \hat{H}^{-1}) \tag{9}$$

For the second issue we use the squared 2-Wasserstein distance, which is well defined for distributions with disjoint support (for instance dirac distributions). So in our case we have:

$$\begin{aligned} W_2(p(w|D), q_s(w))^2 = \\ &= W_2(N(\mu_1, \Sigma_1), N(\mu_2, \Sigma_2)) \\ &= ||\mu_1 - \mu_2||_2^2 + Tr(\Sigma_1 + \Sigma_2 - 2(\Sigma_2^{1/2}\Sigma_1\Sigma_2^{1/2})^{1/2}) \end{aligned} \tag{10}$$

In our case both distributions have the same mean $\mu_1 = \mu_2 = \hat{w}$. The true posterior's covariance matrix is the inverse GGN matrix, i.e. $\Sigma_1 = \hat{H}^{-1}$. For the approximate posterior of the subnetwork we have $\Sigma_2 = \hat{H}_{S+}^{-1}$ which is equal to $\hat{H}_S^{-1}$ (the inverse GGN matrix of the subnetwork) padded with zeros at the positions corresponding to point estimated weights w_r matching the shape of $\hat{H}^{-1}$. We can also define $\tilde{H}_{S+}^{-1} = M_s \odot \tilde{H}^{-1}$, where $\odot$ is the Hadamard produce and M_s is a mask matrix with zeros in the rows and columns corresponding to w_r, i.e. the rows and columns corresponding to weights not included in the subnetwork. Thus we have:

$$\begin{aligned} W_2(p(w|D), q_s(w))^2 = \\ &= W_2(N(\hat{w}, \tilde{H}^{-1}), N(\hat{w}, \tilde{H}_{S+}^{-1}))^2 \\ &= \cancel{||\hat{w} - \hat{w}||_2^2} - Tr(\tilde{H}^{-1} + \tilde{H}_{S+}^{-1} - 2(\tilde{H}_{S+}^{-1/2}\tilde{H}^{-1}\tilde{H}_{S+}^{-1/2})^{1/2}) \\ &= Tr(\tilde{H}^{-1} + \tilde{H}_{S+}^{-1} - 2(\tilde{H}_{S+}^{-1/2}\tilde{H}^{-1}\tilde{H}_{S+}^{-1/2})^{1/2}) \end{aligned} \tag{11}$$

Finding the subset w_s in Eq. (11) would be difficult as the weights depend on each other. Therefore we make an assumption that the weights are independent among each other which results in the following simplified objective:

$$
\begin{aligned}
W_2(p(w|D), q_s(w))^2 = & \\
&= Tr(\tilde{H}^{-1} + \tilde{H}_{S+}^{-1} - 2(\tilde{H}_{S+}^{-1/2}\tilde{H}^{-1}\tilde{H}_{S+}^{-1/2})^{1/2}) \\
&= Tr(\tilde{H}^{-1}) + Tr(\tilde{H}_{S+}^{-1}) - Tr(2(\tilde{H}_{S+}^{-1/2}\tilde{H}^{-1}\tilde{H}_{S+}^{-1/2})^{1/2}) \\
&= \sum_{d=1}^{D} \sigma_d^2 + m_d\sigma_d^2 - 2m_d\sigma_d^2 \\
&= \sum_{d=1}^{D} \sigma_d^2(1 - m_d)
\end{aligned}
\tag{12}
$$

where m_d is the d^{th} diagonal element of M_s, i.e. $m_d = 1$ if w_d is included int he subnetwork or 0 otherwise. As we can see Eq. (12) is minimized by a subnetwork containing the S weights with highest variances. The selection strategy contains variances and not magnitudes as we target predictive uncertainty rather than accuracy.

Step 3: Bayesian Inference. We use the GGN-Laplace approximation to infer a full-covariance Gaussian posterior over the subnetwork's weights $w_s \in \mathbb{R}^S$:

$$
p(w_s|D) \approx q(w_s) = N(w_s; \hat{w}_s, \tilde{H}_S^{-1}) \tag{13}
$$

where $\tilde{H}_S^{-1} \in \mathbb{R}^{S\times S}$ is the GGN w.r.t. the weights w_s:

$$
\tilde{H}_S = \sum_{n=1}^{N} J_{Sn}^{\intercal} H_n J_{Sn} + \lambda_S I \tag{14}
$$

In order to best preserve the magnitude of the predictive variance we udpate the precision to be $\lambda_S = \lambda \cdot S/D$. The weights not belonging in the subnetwork are fixed at their MAP values. Basically we perform full Laplace inference over the selected subnetwork and MAP inference over all remaining weights. Then the resulting approximate posterior will be the following:

$$
q_s(w) = N(w_s; \hat{w}_s, \tilde{H}_S^{-1}) \prod_r \delta(w_r - \hat{w}_r) \text{ from Eq. (13)} \tag{15}
$$

A small issue here is storing and inverting $\tilde{H}_S$, but if the subnetwork is very small it is feasible.

Step 4: Prediction. We perform a local linearization of the NN while fixing w_r to $\hat{w}_r$:

$$
f_{lin}(x, w_s) = f(x, \hat{w}) + \hat{J}_S(x)(w_S - \hat{w}_S) \tag{16}
$$

the corresponding predictive distributions for regressions are:

$$
p(y^\star|x^\star, D) = N(y^\star; f(x^\star, \hat{w}), \Sigma_S(x^\star) + \sigma^2 I) \tag{17}
$$

and for classification:

$$
p(y^\star|x^\star, D) \approx \text{softmax}\left(\frac{f(x^\star, \hat{w})}{\sqrt{1 + \frac{\pi}{8}\text{diag}(\Sigma_S(x^\star))}}\right) \tag{18}
$$

References

Erik Daxberger, Eric Nalisnick, James U Allingham, Javier Antorán, and José Miguel Hernández-Lobato. 2021. Bayesian deep learning via subnetwork inference. In *Proceedings of the 38th International Conference on Machine Learning*.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635* .

Pavel Izmailov, Wesley J Maddox, Polina Kirichenko, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2020. Subspace inference for bayesian deep learning. In *Uncertainty in Artificial Intelligence*. PMLR, pages 1169–1179.

Wesley J Maddox, Gregory Benton, and Andrew Gordon Wilson. 2020. Rethinking parameter counting in deep models: Effective dimensionality revisited. *arXiv preprint arXiv:2003.02139* .

Konstantinos Panousis, Sotirios Chatzis, and Sergios Theodoridis. 2019. Nonparametric Bayesian deep networks with local competition. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*. PMLR, volume 97 of *Proceedings of Machine Learning Research*, pages 4980–4988.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. 2015. Scalable bayesian optimization using deep neural networks. In *International conference on machine learning*. PMLR, pages 2171–2180.