Open Problems for Online Bayesian Inference in Neural Networks

Robert Loftin
NC State University
rtloftin@ncsu.edu

Matthew E. Taylor
Washington State University
taylorm@eecs.wsu.edu

Michael L. Littman
Brown University
mlittman@brown.edu

James MacGlashan
Brown University
james_macglashan@brown.edu

Bei Peng
Washington State University
bei.peng@wsu.edu

David L. Roberts
NC State University
robertsd@csc.ncsu.edu

1 Introduction

Recently, a number of approaches have been developed which allow for approximate Bayesian inference over the parameters of large neural networks. These algorithms can be more robust against overfitting, and can provide useful estimates of model uncertainty. In domains such as active learning Golovin et al. (2010) and Bayesian reinforcement learning (Vlassis et al., 2012), an ideal learner needs to not only estimate the posterior given the current training data, but also needs to reason about how new observations might affect that posterior, so that new data can be collected as efficiently as possible. We argue that existing *batch* techniques for Bayesian neural networks are not well suited to such *online* inference, and in this preliminary work we discuss ways in which these algorithms can be adapted to allow for efficient online inference.

If we view the active learning problem as a special case of the Bayesian reinforcement learning problem, we can see that selecting new instances requires searching through a set of unlabeled instances, and evaluating the quality of the posterior when new instances are added to the training data. We therefore require algorithms which can efficiently update the current posterior to take into account new instances, and which accurately estimate the change in the posterior resulting from the new data. Here we describe a potential approach to this problem, in which a variational estimate of the posterior is represented by two sets of parameters, one of which is used to estimate the prior based on the current data, and the other used to perform online updates with *hypothetical* data. We also describe how contextual information about the learning task may be used to find posterior approximations that are better suited to online inference. The main goal of this work however is to motivate further research into the problems of online inference in Bayesian neural networks.

2 Interactive Learning

To motivate this work, we consider a hypothetical interactive learning problem in which one or more human teachers provide labels y for instances x selected by the learner. The learner is given some initial labeled data D and a prior $p(\hat{w})$ over the connection weights, accumulated in $\hat{w}$, for a neural network that is assumed to have generated the labels. The learner is then presented with a set C of unlabeled instances, drawn from some unknown joint distribution $r(C)$, and will be asked to provide a label some for randomly selected $x \in C$. Before labeling x, the learner can select a number of other instances from C and ask the human teacher to provide labels for these instances.

We call the set C a *context*, and we assume that certain instances are more likely to co-occur within the same context. For example, a household robot, being taught recognize objects, might first be taught about objects in the kitchen, and then in the living room, such that the robot can only request labels for objects in its immediate vicinity. Training data for an image classifier might be collected

Workshop on Bayesian Deep Learning, NIPS 2016, Barcelona, Spain.

from a number of users, each of whom provides a separate set of images, and only provides labels for their own images.

Ideally, the learner should first compute the batch posterior $p(\hat{w}|D)$, and then compute the online posterior $p(\hat{w}|D, o)$ for each possible observation o, where an observation is a set of pairs (x_i, y_i) of instances in the context C, along with possible labels for those instances. The learner evaluates each posterior in terms of the expected accuracy of the resulting label for the target x under that posterior, and chooses the set of instances that maximizes the expected accuracy over different possible labelings. Where only approximate inference is possible, the performance of the learner will depend on the accuracy of the batch and online posterior approximations, and in particular on how well the approximation captures the dependencies between instances that are likely to occur in the same context. We will see that, when some information about context is available in the initial data D, it is possible to incorporate that knowledge into the batch approximation to improve the quality of the online approximations.

3 Online vs. Batch Inference

For this interactive learning problem, we first find an approximation of the batch posterior conditioned on all the existing training data, and then approximate a large number of online posteriors given potential observations. Existing variational approaches based on optimization Gal and Ghahramani (2015) are in general not well suited to online inference because 1) estimating the posterior given a single new data point requires optimization over the entire data set and 2) the influence of any single data point on the the posterior may be small, and so may not be adequately captured by the global approximation. Approaches based on Assumed Density Filtering (AFD) and Expectation Propagation (EP) (Minka, 2001) do allow for online updates, as the update for a single data point explicitly estimates the posterior given that point, rather than simply adjusting the approximation in the direction of that posterior. While AFD techniques such as probabilistic backpropagation (PBP) (Hernández-Lobato and Adams, 2015) and expectation backpropagation (EBP) (Soudry et al., 2014) can be applied simultaneously to estimate both the batch and online posteriors, using AFD for batch inference may not always be desirable, as AFD relies on approximations that my not be accurate for some network representations, and may not converge when applied repeatedly through Expectation Propagation.

As online and batch inference have very different requirements, it makes sense to choose a framework in which the algorithm used for online inference can be selected and tuned independently of that used for batch inference. We can define a variational distribution $q(\hat{w}; \phi, \theta)$, where θ are the parameters of the batch posterior (encoding the existing data D), while ϕ are the parameters of the online posterior (encoding a new observations o). We define a set of default online parameters $\bar{\phi}$, such that the batch posterior is represented by $q(\hat{w}; \bar{\phi}, \theta)$. For each potential observation o, we can compute parameters ϕ_o, of the approximate online posterior $q(\hat{w}; \phi_o, \theta)$. With this representation, we can use very different algorithms for finding θ and ϕ.

As a concrete example, we can represent the weight distribution of a connection between nodes i and j as $w_{ij} \sim \mathscr{N}(\theta_{ij} + \phi^{\mu}_{ij}, \theta^2_{ij}\phi^{\nu}_{ij})$, where ϕ^{μ} are the online means and ϕ^{ν} are the online variances, such that all w_{ij} are independent. If we set $\bar{\phi}^{\mu} = \mathbf{0}$ and $\bar{\phi}^{\nu} = \mathbf{1}$, then we can see that the batch posterior has the same form as that learned by DropConnect Wan et al. (2013), with Gaussian noise, while the online posterior is of the form used by probabilistic backpropagation (PBP). It has been shown that dropout training algorithms can be seen as a form of variational inference Gal and Ghahramani (2015), and so we can use backpropagation with DropConnect to learn θ for the batch posterior $q(\hat{w}; \bar{\phi}, \theta)$. We can then use PBP to estimate the parameters ϕ_o of the online approximations for each possible observation o, starting with $q(\hat{w}; \bar{\phi}, \theta)$ as the prior.

4 Alternative Variational Objective

The use of the batch posterior as a starting point for subsequent online inference suggests that the standard variational objective $\mathrm{D_{KL}}(q(\hat{w}; \bar{\phi}, \theta) \| p(\hat{w}|D))$ may not always be the best choice for learning the batch approximation, for two main reasons. First, the batch posterior defined by θ may conflict with the approximations used for online inference. The online parameters might only be applied to a subset of the connection weights, in which case, for certain values of θ the connections covered by ϕ may not have a strong influence on the output of the network, such that the approximate online posteriors will not differ significantly from the batch posterior. Second, the variational approximation may not capture the most important dependencies between the outputs of the network for different instances. This is where the idea of context becomes important, as it is more important to capture covariances between instances that are likely to appear in the same context. Similarly, context can help resolve the first issue of mismatched approximations, by indicating which online updates need to be the most accurate.

If the initial data set D includes contextual information, we can design an objective function for θ that takes into account the importance of accurately capturing the relationships between instances in the same context. We assume that the initial data D is divided into M sets D_i, which we refer to as *contexts*, each composed of $N_i \ll M$ instances x_{ij} and labels y_{ij}. Furthermore, we assume that the instances in each set D_i are drawn from the context distribution $r(C)$ that will be used during online learning. We wish to find θ such that the approximate online posterior $q(\hat{w}; \phi_o, \theta)$ accurately estimates the label distribution for instance x, when x and the instances of observation o are found in the same context. To do this, we can divide each context D_i into a training set D_i^{train} and test set D_i^{test}, and define the variational objective function as

$$F(\theta; D^{test}, D^{train}) = \sum_i \int_{\hat{w}} \Bigg[q(\hat{w}; \bar{\phi}, \theta) \sum_{j \in D_i^{train}} \log p(y_{ij}|x_{ij}, \hat{w}) + q(\hat{w}; \phi_\theta^i, \theta) \sum_{j \in D_i^{test}} \log p(y_{ij}|x_{ij}, \hat{w}) \Bigg] d\hat{w} - \mathrm{D}_{\mathrm{KL}}[q(\hat{w}; \bar{\phi}, \theta) \| p(\hat{w})],$$

where ϕ_θ^i are the parameters of the approximations of $p(\hat{w}|D^{train}, D_i^{test})$ computed using the online update algorithm. This objective function balances between the quality of $q(\hat{w}; \phi, \theta)$ as an approximation of $p(\hat{w}|D^{train})$, and the quality of $q(\hat{w}; \phi_\theta^i, \theta)$ as an approximation of $p(\hat{w}|D^{train}, D_i^{test})$. In doing so, it sacrifices some of the data D, and some degree of accuracy of the batch posterior $q(\hat{w}; \bar{\phi}, \theta)$ to improve the predictive accuracy of the online posteriors $q(\hat{w}; \phi_o, \theta)$. The the specific algorithm for the optimization of θ under this objective will depend heavily on the algorithm used for online inference, as the online parameters ϕ_θ^i may not be differentiable with respect to θ.

5 Conclusion

Many of the most interesting applications of Bayesian methods depend on the ability to reason about changes in a learner's belief, both in response to new observations, and in response to possible actions the learner could take. Here we have discussed some of the difficulties that arise in applying Bayesian neural networks to applications requiring such online inference, including the limitations of variational methods, and the risks of choosing poor approximations of batch posteriors. The algorithmic framework described here provides a potential starting point for future work on these issues, and more importantly, illustrates some of the specific areas in which inference techniques can be improved with respect to online applications. This work highlights the need for variational algorithms which explicitly consider how their posterior estimates will be used, and the value of techniques that allow large numbers of related online updates to be performed efficiently. At the same time however, this work suggests that extremely difficult inference problems such as Bayesian reinforcement learning may be feasible even with with large neural networks.

References

Gal, Y. and Ghahramani, Z. (2015). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *arXiv preprint arXiv:1506.02142*.

Golovin, D., Krause, A., and Ray, D. (2010). Near-optimal bayesian active learning with noisy observations. In *Advances in Neural Information Processing Systems*, pages 766–774.

Hernández-Lobato, J. M. and Adams, R. P. (2015). Probabilistic backpropagation for scalable learning of bayesian neural networks. *arXiv preprint arXiv:1502.05336*.

Minka, T. P. (2001). *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology.

Soudry, D., Hubara, I., and Meir, R. (2014). Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights. In *Advances in Neural Information Processing Systems*, pages 963–971.

Vlassis, N., Ghavamzadeh, M., Mannor, S., and Poupart, P. (2012). Bayesian reinforcement learning. In *Reinforcement Learning*, pages 359–386. Springer.

Wan, L., Zeiler, M., Zhang, S., Cun, Y. L., and Fergus, R. (2013). Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1058–1066.