Dropout-based Automatic Relevance Determination

Dmitry Molchanov[1], **Arseniy Ashuha**[2] **and Dmitry Vetrov**[3,4*]
[1]Skolkovo Institute of Science and Technology
[2]Moscow Institute of Physics and Technology
[3]National Research University Higher School of Economics, [4]Yandex
Moscow, Russia
[1]`dmitry.molchanov@skolkovotech.ru`, [2]`ars.ashuha@phystech.edu`, [3]`vetrovd@yandex.ru`

1 Introduction

During past several years, several works about treating dropout as a Bayesian model have appeared [1,3,4]. Authors of [1] proposed a method that represents dropout as a special case of Bayesian model and allows to tune individual dropout rates for each weight.

Another direction of research is the exploration of the ways to train neural networks with sparse weight matrices [6,9] and the ways to prune excessive neurons [8]. One of the possible approaches is so-called Automatic Relevance Determination (ARD). A typical example of such models is the Relevance Vector Machine [4]. However, these models are usually relatively hard to train.

In this work, we introduce a new dropout-based way to obtain the ARD effect that can be applied to complex models via stochastic variational inference framework.

2 Theory

Variational dropout [1] is a recent technique that generalizes Gaussian dropout [7]. It allows to set individual dropout rates for each weight/neuron/layer and to tune them with a simple gradient descent based method. As in most modern Bayesian models, the goal is to obtain the variational approximation to the posterior distribution via optimization of the variational lower bound (1):

$$\mathcal{L}(q) = \mathbb{E}_{q(w)} \log p(T \,|\, X, w) - D_{KL}(q(w) \,\|\, p_{prior}(w)) \to \max_q \tag{1}$$

To put it simply, the main idea of Variational dropout is to search for posterior approximation in a specific family of distributions: $q(w_i) = \mathcal{N}(\theta_i, \alpha_i \theta_i^2)$. Then the reparameterization trick [12] for this model is equivalent to plain Gaussian dropout $w_i \sim q(w_i) \Leftrightarrow w_i = \theta_i(\sqrt{\alpha_i}\varepsilon_i + 1)$, $\varepsilon_i \sim \mathcal{N}(0,1)$. Now we can treat $\boldsymbol{\alpha}$ as variational parameters and optimize the lower bound w.r.t. $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$. The prior distribution is chosen to be log-scale uniform distribution. If we optimize the lower bound w.r.t $\boldsymbol{\theta}$ with fixed $\boldsymbol{\alpha}$, log-scale uniform prior makes Variational dropout equivalent to Gaussian dropout.

In the original paper, authors reported difficulties in training the model with large values of dropout rates $\boldsymbol{\alpha}$ and only considered the case of $\alpha_i \leq 1$ (dropout rate 0.5 or less). However, the case of large α_i is very interesting as having high dropout rates ($\alpha_i \to +\infty$) effectively means that i-th node, i.e. a weight or a neuron is always ignored and can be removed from the model. We study this case in a simpler setting of linear models, such as linear regression and binary and multiclass logistic regression. Because of the limited space, we presented only multiclass logistic regression results in the experimental section.

[*] Bayesian Methods Research Group: `http://bayesgroup.ru`

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

Table 1: Test set prediction accuracy and achieved sparsity level

	Accuracy			Sparsity		
Dataset	VD-ARD	L1-LR	RVM	VD-ARD	L1-LR	RVM
MNIST	**0.926**	0.919	N/A	**69.8%**	57.8%	N/A
DIGITS	0.948	**0.955**	0.945	**75.4%**	38.0%	74.6%
DIGITS + noise	0.930	**0.937**	0.846	**87.6%**	55.9%	85.7%

3 Contribution

Our contribution consists of three parts. Firstly, we came up with a new approximation of the KL divergence (2). As KL divergence is not tractable [1], we used a sampling technique to obtain the approximation (2). The original estimate was obtained under the assumption that $\alpha_i \leq 1$ [1], so we propose an estimation of the KL divergence that remains correct for all values of $\boldsymbol{\alpha}$. An illustration of different approximations is presented in figure 1.

$$-D_{KL}(q(w_i\,|\,\theta_i,\alpha_i)\|p_{prior}(w_i)) \approx 0.64\,\sigma(1.5\,(1.3+\log\alpha_i)) - 0.5\,\log(1+\alpha_i^{-1})) + \text{const} \tag{2}$$

Secondly, we came up with a trick that allows us to train this model in a stochastic setting and to learn large values of $\boldsymbol{\alpha}$, avoiding the problem of gradients with large variance. We found that using another parameterization is crucial for training such models. We introduce $\sigma_i^2 = \alpha_i\theta_i^2$ and optimize the lower bound w.r.t. $(\boldsymbol{\theta}, \log\boldsymbol{\sigma}^2)$ instead of $(\boldsymbol{\theta}, \boldsymbol{\alpha})$.

$$w_i \sim q(w_i\,|\,\theta_i,\alpha_i) \Leftrightarrow w_i = \theta_i(1+\sqrt{\alpha}\cdot\epsilon_i) = \theta_i + \sigma_i\cdot\epsilon_i, \;\; \epsilon_i \sim \mathcal{N}(0,1) \tag{3}$$

This trick transforms the multiplicative noise of Gaussian dropout to an equivalent additive noise. The loss function remains the same. However, this method radically reduces the variance of the gradient w.r.t. $\boldsymbol{\theta}$, because the additive noise does not depend on $\boldsymbol{\theta}$.

Finally, we showed explicitly that driving the dropout rate α_i to infinity makes the variational distribution $q(w_i)$ tend to a delta function centered at 0, which makes our model similar to classical ARD models. However, it should be noted that this approach is different. In dropout-based models, the ARD effect arises when we put an infinitely strong noise on model weights through approximate Bayesian inference in a model with a fixed prior. In classical ARD models [4] the prior distribution is not fixed and is tuned on the training dataset. Our approach comes more naturally from the Bayesian framework, as the prior distribution should remain independent from the training data.

An iteration of the final training scheme consists of taking a minibatch, sampling the activations using the local reparameterization trick [1], computing an estimate of the lower bound (2) and its gradient using parametrization trick (3) $(\boldsymbol{\theta}, \log\boldsymbol{\sigma}^2)$. This stochastic gradient is then used to update model parameters via some SGD-based optimization method (we used Adam [2]).

4 Experiments

We have explored the relevance determination quality of our algorithm as well as the classification accuracy of the resulting sparse model. We used MNIST [10] and DIGITS [11] datasets for evaluation. We compared our approach to other standard methods of training sparse linear models, such as L1 logistic regression (L1-LR) and the Relevance Vector Machine classifier (RVC). The percentage of removed features and the test set classification accuracy can be seen in Table 1. The result of the RVC on MNIST is not present, as most available implementations of the RVC can not work with datasets of that size (60000 objects in the training dataset).

The optimal value of the regularization parameter for L1-LR was found using cross-validation. The RVC was used with default hyperparameters and our method does not have any hyperparameters.

We clipped the result by the value $\log\alpha_i \geq 5$, which corresponds to binary dropout rate greater than 0.99. Corresponding parameters θ_i were set to zero during testing time in all experiments.

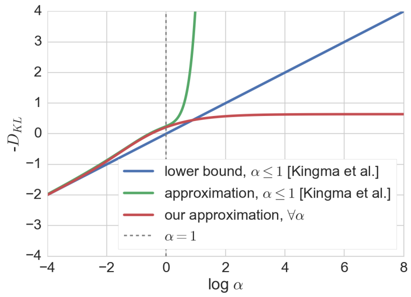

Figure 1: An illustration of the different approximations of divergence.

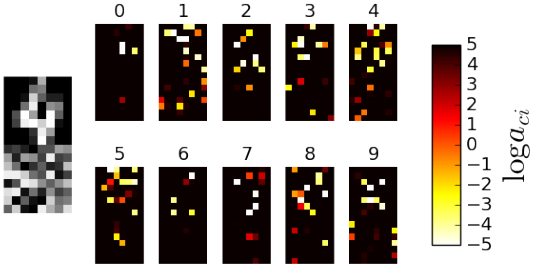

Figure 2: Left: an example of an object with concatenated noise. Right: trained dropout rates, each heat map corresponds to a weight vector in multiclass model. c is a class label and i is a feature index

We also experimented with introducing irrelevant features to DIGITS dataset. We concatenated each picture with a picture of the same size that contains independent Gaussian noise of roughly the same magnitude as the original image. An example of an object and the resulting dropout rates are shown in Figure 2. The code is available at `https://github.com/da-molchanov/vd-ard-bdl16`.

5 Discussion

We showed that the ARD effect is quite strong, and the accuracy of the resulting model is relatively high despite the fact that we used stochastic optimization instead of efficient and accurate solvers that are used in L1-regression or the RVM. Our future work would be to apply this approach to neural networks because this method could lead to discovering a way of training sparse neural networks. It could also be used as a regularization technique or as a way to prune excessive weights or groups of weights from the model, which in fact allows adjusting network topology automatically.

Acknowledgments

This work was supported by RFBR project No. 15-31-20596 (mol-a-ved) and by Microsoft: Moscow State University Joint Research Center (RPD 1053945).

References

[1] Kingma, D.P. & Salimans, T. & Welling, M. (2016). Variational Dropout and the Local Reparameterization Trick. In C. Cortes and N. D. Lawrence and D. D. Lee and M. Sugiyama and R. Garnett (eds.). *Advances in Neural Information Processing Systems 28*, pp. 2575–2583. Curran Associates, Inc. Press.

[2] Kingma, D.P. & Ba J. (2014). Adam: A Method for Stochastic Optimization. *The International Conference on Learning Representations (ICLR)*, San Diego, 2015.

[3] Gal, Y. & Ghahramani, Z. (2015). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning*.

[4] Tipping, M. E. (2001). Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research* 1, pp. 211–244.

[5] Maeda, X.-i. (2014). A Bayesian encourages dropout. *arXiv preprint arXiv:1412.7003*.

[6] Louizos C. (2015). Smart Regularization of Deep Architectures. *Master Thesis*.

[7] Nitish Srivastava & Geoffrey Hinton & Alex Krizhevsky & Ilya Sutskever & Ruslan Salakhutdinov (2014). *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, Journal of Machine Learning Research 15, pp. 1929-1958.

[8] Karaletsos, T. & Rätsch, G. (2015). Automatic Relevance Determination For Deep Generative Models. *arXiv preprint arXiv:1505.07765*.

[9] Scardapane, S. & Comminiello, D. & Hussain, A. & Uncini, A. (2016). Group Sparse Regularization for Deep Neural Networks. *arXiv preprint arXiv:1607.00485*.

[10] LeCun, Y. & Bottou, L. & Bengio, Y. & Haffner, P. (1998). Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 86(11):2278-2324.

[11] Lichman, M. (2013). UCI Machine Learning Repository [`http://archive.ics.uci.edu/ml`]. Irvine, CA: University of California, School of Information and Computer Science.

[12] Kingma, D. & Welling, M. (2014). Auto-Encoding Variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations*.