Variational Hypernetworks

Agustinus Kristiadi
University of Bonn
kristiadi@uni-bonn.de

1 Motivation

- Matrix variate normal (MVN) [Gupta and Nagar, 1999] is a popular choice of variational posterior of neural network (NN)'s weights. It has been used by recent methods for bayesian neural networks [Louizos and Welling, 2016, Sun et al., 2017, Zhang et al., 2017]. MVN is parameterized by mean matrix and two Kronecker factors of the covariance. That is $\mathbf{X} \sim \mathcal{MVN}(\mathbf{X} \mid \mathbf{M}, \mathbf{A}, \mathbf{B}) \iff \text{vec}(\mathbf{X}) \sim \mathcal{N}(\text{vec}(\mathbf{X}) \mid \text{vec}(\mathbf{M}), \mathbf{A} \otimes \mathbf{B})$.
- Variational autoencoder (VAE) [Kingma and Welling, 2013, Rezende et al., 2014] meanwhile, is a popular latent variable method, which models the posterior of NN's hidden units. It uses inference network to parameterize the factorial Gaussian posterior of hidden units.
- **Idea:** Can we instead use inference network to parameterize the MVN posterior of NN's weights?
  - By doing so, hypothetically we enjoy the power of NN to represent the posterior, while can still benefit from our familiarity with Gaussian, e.g. tractability.
  - From different perspective, we can see this as a Bayesian version of hypernetworks [Ha et al., 2016].
- This is in contrast to: (i) directly doing variational inference (VI) on MVN posterior's native parameters [Louizos and Welling, 2016, Sun et al., 2017], (ii) by using some implicit distributions [Pawlowski et al., 2017, Krueger et al., 2017, Louizos and Welling, 2017], or (iii) by using simple Laplace approximation [Ritter et al., 2018] on the MAP estimate. See Section 3 for the complete discussion.
- **Hypothetical justification:**
  - Our proposed method should perform better than methods in (i) as our posterior is more expressive due to the usage of the inference network.
  - For point (ii), as we use MVN which is tractable, we know how to both sample and evaluate the posterior in closed form, whereas in implicit models, we can only sample.
  - Lastly for point (iii), Laplace approximation only captures the local covariance around the mode, so it might not capture the global properties of the posterior. Meanwhile our proposed method is a full variational method, so we can better capture the global structure of the posterior.

2 Proposed approach

- **Notations:**
  - $\mathbf{X}$ is the input.
  - $\mathbf{W}_l$ is the l-th layer weight matrix.
  - $\mathbf{h}_l$ is the l-th layer hidden units.
  - $\mathbf{y}$ is the outputs.
  - σ is pointwise nonlinearity.

Preprint. Work in progress.

- L is the number of NN layers.

- **Assumptions:**
  - Multi layer perceptron (MLP) architecture.
  - Diagonal MVN variational posterior on each weight matrix.
  - The MVN variational posterior's mean matrix and both factor matrices are parameterized by a neural network.

- **Model:**
  - Denote $\mathbf{W}$ as the concatenation of all $\mathbf{W}_l$ for all $l = 1 \dots L$.
  - Let $\mathbf{a}_{i<j} := \mathbf{a}_1, \dots, \mathbf{a}_{j-1}$.
  - The graphical model is a chain.
  - Denote $\mathbf{h}_0 := \mathbf{X}$ and $\mathbf{h}_L := \mathbf{y}$.

$$
\begin{aligned}
p(\mathbf{W}) &:= p(\mathbf{W}_1 \,|\, \mathbf{X}) p(\mathbf{W}_2 \,|\, \mathbf{h}_1, \mathbf{W}_1, \mathbf{X}) \dots p(\mathbf{W}_L \,|\, \mathbf{h}_{l<L}, \mathbf{W}_{l<L}, \mathbf{X}) && (1)\\
&= p(\mathbf{W}_1 \,|\, \mathbf{X}) p(\mathbf{W}_2 \,|\, \mathbf{h}_1), \dots, p(\mathbf{W}_L \,|\, \mathbf{h}_{L-1}) && (2)\\
&= \prod_{l=1}^{L} p(\mathbf{W}_l \,|\, \mathbf{h}_{l-1}) && (3)
\end{aligned}
$$

- **Variational posterior:**
  - The prior is $\mathcal{MVN}(\mathbf{W}_l \,|\, \mathbf{0}, \mathbf{I}, \mathbf{I}) \;\; \forall\, l = 1 \dots L$.
  - $q_l(\mathbf{W}_l \,|\, \mathbf{h}_{l-1}, \boldsymbol{\phi}_l) := \mathcal{MVN}(\mathbf{W}_l \,|\, \mathbf{M}_l, \text{diag}(\mathbf{a}_l), \text{diag}(\mathbf{b}_l))$ is the l-th layer posterior.
  - $f_l(\mathbf{h}_{l-1}; \boldsymbol{\phi}_l)$ is the inference network modeling $\mathbf{M}_l, \mathbf{a}_l, \mathbf{b}_l$ with parameters $\boldsymbol{\phi}_l$.

- **Generative process:**
  - Note at layer L, σ is the identity map.
  - For each layer $l = 1 \dots L$:
    - $\mathbf{M}_l, \mathbf{a}_l, \mathbf{b}_l = f_l(\mathbf{h}_{l-1}; \boldsymbol{\phi}_l)$.
    - $\mathbf{W}_l \sim \mathcal{MVN}(\mathbf{W}_l \,|\, \mathbf{M}_l, \text{diag}(\mathbf{a}_l), \text{diag}(\mathbf{b}_l))$.
    - $\mathbf{h}_l = \sigma(\mathbf{h}_{l-1}\mathbf{W}_l)$.

- **Learning:**
  - Let $q(\mathbf{W} \,|\, \mathbf{h}, \boldsymbol{\phi}) := \prod_{l=1}^{L} q_l(\mathbf{W}_l | \mathbf{h}_{l-1}, \boldsymbol{\phi}_l)$, where $\boldsymbol{\phi}$ is the concatenation of $\boldsymbol{\phi}_l$ for all $l = 1 \dots L$.
  - Minimize negative evidence lower bound (ELBO) objective:
    $\mathcal{L}(\boldsymbol{\phi}) = \lambda \sum_{l=1}^{L} D_{\text{KL}}[q_l(\mathbf{W}_l \,|\, \mathbf{h}_{l-1}, \boldsymbol{\phi}_l) \,\|\, \mathcal{MVN}(\mathbf{0}, \mathbf{I}, \mathbf{I})] - \mathbb{E}_q[p(\mathbf{y} \,|\, \mathbf{X}; \mathbf{W}]$.

2.1 Reducing parameter count

A couple of approaches have been proposed to reduce the parameter count of a hypernetwork. First, Ha et al. [2016] parametrize the (main NN) weight $\mathbf{W}$ by a matrix $\mathbf{V}$ and a vector $\mathbf{d}$. If the weight matrix is in $\mathbb{R}^{r \times c}$, then $\mathbf{V}$ is also in $\mathbb{R}^{r \times c}$, while $\mathbf{d}$ is in $\mathbb{R}^r$. Then,

$$
\mathbf{W} = \begin{bmatrix} d_1 \mathbf{v}_1 \\ d_2 \mathbf{v}_2 \\ \dots \\ d_r \mathbf{v}_r \end{bmatrix} \quad . \tag{4}
$$

That is, each element of $\mathbf{d}$ is used as the scaling for each row of the matrix $\mathbf{V}$ to produce $\mathbf{W}$, which implies we are only able to express $\mathbf{W}$ with less degree of freedom. However, the crucial advantage is that only $\mathbf{d}$ comes from hypernetwork, thus we have more efficient parametrization compared to the hypernetwork with the full matrix as the output. Therefore, this approach is a compelling trade-off option between expressiveness and computational cost.

Similarly to Ha et al. [2016], Krueger et al. [2017] proposed to use vector scaling $\mathbf{d}$ on a matrix $\mathbf{V}$. The main distinction is that now the matrix is row-normalized. Thus this is analogous to

Model	#params
Vanilla & Dropout	79,400
MVN	80,394
Naive hypernet	2,518,794
Hypernet with vector scaling	164,198

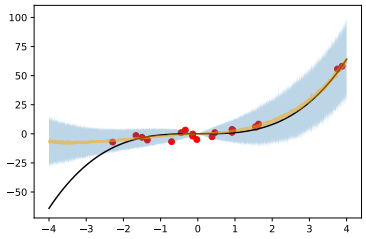
Table 1: Number parameters of various models on small MLP of size 784-100-10, without bias for simplicity. The hypernetworks' hidden layer size is 30, if applicable.

weight normalization scheme [Salimans and Kingma, 2016]. All in all, the expressiveness of the parametrization of this method is similar to that of Ha et al. [2016].

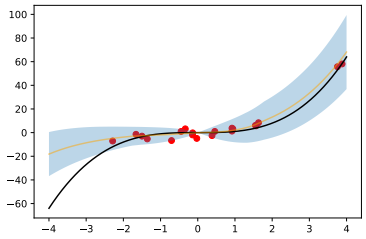
Similar to both approaches, we propose to use vector scaling parametrization indirectly, i.e. we scale the mean matrix $\mathbf{M}$ of the variational posterior, instead of scaling the weight matrix directly. We compile the comparison of these approaches with the other parametrizations, presented in Section 2.1.

3 Related work

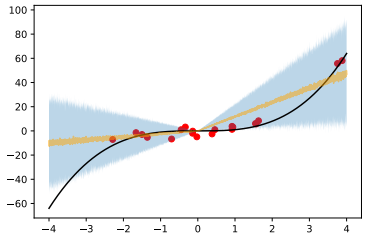
- Louizos and Welling [2016] and Sun et al. [2017] proposed to use diagonal MVN as the variational posterior for each layer in an NN. The variational parameters are one mean matrix and two factor matrices per layer. In contrast, our proposed method's variational parameters are the parameters of the neural networks.
- Zhang et al. [2017] uses Kronecker factored Fisher information matrix (FIM) as the covariance of MVN variational posterior. Their main contribution is the learning algorithm, called Noisy K-FAC [Martens and Grosse, 2015], as it is derived by applying Natural Gradient (NG) on ELBO objective.
- Blundell et al. [2015] proposed Bayes-by-Backprop (BbB) algorithm which models the weights of an NN with diagonal Gaussian. As in Louizos and Welling [2016], BbB does not use inference network to parametrize the variational posterior.
- Hernández-Lobato and Adams [2015] proposed to use fully-factored Gaussian posterior combined with Expectation Propagation method [Minka, 2001], resulting to a method called probabilistic backpropagation.
- Our proposed work, meanwhile is combining the power of neural network with tractable posterior: we use inference network to parameterize MVN variational posterior. It very similar to VAE [Kingma and Welling, 2013, Rezende et al., 2014], as we have similar architecture, though we focus on the posterior over weights instead of over hidden units.
- From different perspective, our proposed method is a Bayesian version of hypernetwork [Ha et al., 2016]. That is, we use hypernetwork as the inference network to parameterize the variational posterior.
- Krueger et al. [2017] also proposed a Bayesian version of hypernetwork. However they use an implicit distribution as their variational posterior. In contrast, we still use tractable variational posterior, thus we can still evaluate the likelihood of the variational posterior in closed form. This work does not consider the full posterior of the weight, but instead only considers the vector factor of the normalized weight matrix [Salimans and Kingma, 2016]. Thus their method is efficient, even though they sacrifice the expressiveness of the variational posterior.
- Pawlowski et al. [2017] proposed to use hypernetwork to parameterize implicit distribution, resembling generator network in GAN. It differs from Krueger et al. [2017] in the choice of the model and the approximation of the KL-divergence term in the ELBO objective.
- Louizos and Welling [2017] proposed Multiplicative Normalizing Flow (MNF). The main idea is use auxiliary random variable $\mathbf{z}$ to formulate the variational posterior in the form of $q_l(\mathbf{W}) = \int_{\mathcal{Z}} q_l(\mathbf{W} \mid \mathbf{z}) \mathrm{d}p(\mathbf{z})$, i.e. a mixture distribution. Thus, although they choose $q_l(\mathbf{W} \mid \mathbf{z})$ to be MVN, $q_l(\mathbf{W})$ can represent complicated distribution. The latent variable $\mathbf{z}$ is transformed with a variant of normalizing flow, i.e. MNF, to further enhance the



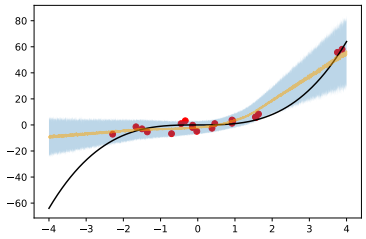
(a) MVN hypernet naive (ours)



(b) MVN hypernet with vector scaling (ours)



(c) MVN



(d) Dropout

Figure 1: Various approaches applied on the toy regression problem [Hernández-Lobato and Adams, 2015]. The black line is the true (noiseless) function, orange line is the mean, and shaded region is the ± 3 standard deviation of the model's predictive distribution.

expressiveness of $q_l(\mathbf{W})$. The trade-off of this approach is that the variational posterior is now intractable and MNF must resort on approximation when computing the KL-divergence term of the ELBO. In contrast, our proposed approach uses arbitrary neural network instead of normalizing flow, and uses MVN variational posterior so it is tractable.

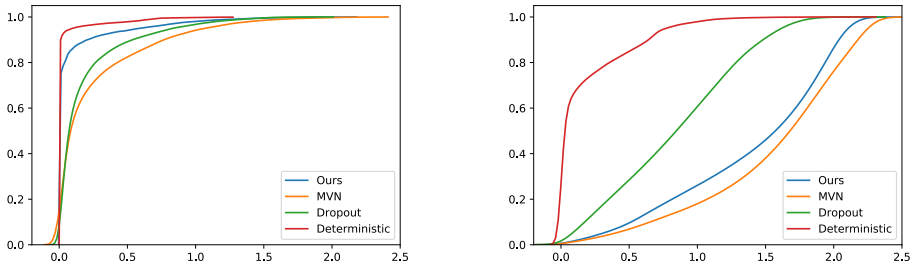
- Ritter et al. [2018] proposed to use simple, cheap, and scalable approximation of the posterior. The method can be summarized in two step: MAP estimation of weights and Laplace approximation [MacKay, 1992] of the posterior around the MAP estimate. The covariance of the Laplace approximation is obtained by using Kronecker factored FIM, thus they assumed block diagonal structure on the FIM. This FIM can be approximated efficiently using running average of minibatch statistics. Though fast, simple and can be applied on any MAP-trained NNs, Laplace approximation can only capture the local structure around the MAP estimate, and might not be able to capture the global property well.

4 Preliminary result

The results presented in this section were produced using almost identical hyperparameters. That is, we optimized all models for 20k iterations, batch size of 200, using default parameter of Adam. The only hyperparameter that we tuned was the regularization strength: the KL-term and the L^2-term on ELBO and MAP objective respectively. For MNIST dataset specifically, we tried out all values in $\{10^{-n}\}_{n=1}^{5}$, and pick the one which balances uncertainty on notMNIST while still maintaining reasonable accuracy on MNIST (i.e. at least achieve 95% accuracy on MNIST test set). Finally, the predictive distribution is approximated by averaging over 100 variational posterior samples.

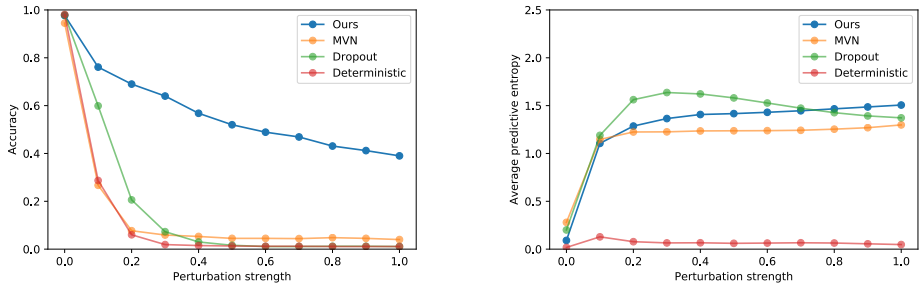
4.1 Toy regression

We consider the toy regression problem [Hernández-Lobato and Adams, 2015]. The model is MLP of size 1-100-1, along with 10 hidden units on each hypernetwork. Note that we do not use hypernetwork in deterministic, dropout, and (non-hypernetwork) MVN [Louizos and Welling, 2016] settings. The results are presented in Figure 1. As can be seen above, the mean of the predictive distribution is closer



(a) MNIST (b) notMNIST

Figure 2: Preliminary results of the CDF of the predictive entropy on MNIST with small MLP.



(a) Accuracy (b) Entropy

Figure 3: Accuracy and average entropy of various methods when attacked by FGSM with varying perturbation strength.

to the true function, even though we have only few data points, compared to the non-hypernetwork MVN posterior, the dropout, and deterministic model.

Furthermore, parametrizing the variational posterior with vector scaling does not seem to have any ill effect in this case, compared to the naive parametrization. Thus this provides a justification for our proposed approach. Therefore from now on, we will only consider the vector scaling parametrization.

4.2 Out-of-distribution uncertainty

We did an experiment on MNIST similar to Louizos and Welling [2017], Ritter et al. [2018]: we trained our model as usual with MNIST training set and observed the uncertainty on both MNIST test set and notMNIST [1] test set. Hypothetically, a good model would predict with high accuracy and low uncertainty (high confidence) on MNIST test set, while reporting high uncertainty (low confidence) on notMNIST test set, as notMNIST data are not identically distributed to MNIST data. We use an MLP of size 784-100-10 in this experiment. If applicable, we use 50 hidden units for the hypernetworks.

The results are presented in Figure 2. We follow Louizos and Welling [2017] by plotting the CDF of the entropy of the predictive distribution w.r.t. both test sets. A good models will have a CDF that is closer to the top left on MNIST, implying it has low uncertainty; while having a CDF that is closer to the bottom right on notMNIST, implying it has high uncertainty. We observe that our proposed approaches, attains competitive results.

[1] Available at http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html.

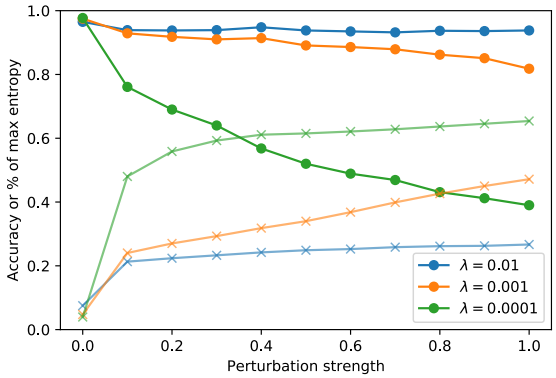


Figure 4: Accuracy and average entropy of our proposed model, with varying regularization strength λ. Circle indicates accuracy, while cross indicates entropy.

4.3 Adversarial examples

To measure the robustness of our proposed method to adversarial examples [Szegedy et al., 2013, Goodfellow et al., 2014], we apply Fast Gradient Sign Method (FGSM) [Goodfellow et al., 2014] on MNIST test set w.r.t. our proposed approach and the baselines. The results are presented in Figure 3. We observe that our method is able to at least match the uncertainty produced by the baselines, while significantly more robust (in term of accuracy) to the adversarial attack.

It is also interesting how the regularization strength influence our proposed method. To answer this question, we experimented with several values of λ and plot each of the accuracy and the entropy in Figure 4. We can observe that λ can be seen as trade-off hyperparameter to control between robustness and higher uncertainty. Observe also that using $\lambda = 0.01$, we achieve almost no drop in the accuracy, even when the perturbation strength is at the maximum.

5 Next step

- We propose to use diagonal MVN. Would it be possible to use more fine-grained covariance structure while keeping the complexity in check? E.g. model the covariance factors $\mathbf{A}_l, \mathbf{B}_l$ as rank one matrices.
- Explicitly drawing samples of weights can be very expensive as we draw one weight matrix per layer per data point. Thus each layer's sampled weights is a 3-dimensional tensor. Instead we can use local reparametrization trick [Kingma et al., 2015] to instead draw samples of $\mathbf{h}_{l+1}$ directly which is a matrix of usually much lower dimensional.
- There seems to be a connection between the size of the inference network and the performance, especially in the adversarial attack experiment. It might be instructive to experiment further on this.
- It is interesting to apply the idea to convolutional neural networks (CNNs). We can assume that the kernels in each layers are i.i.d, which reduces the number of parameter further.
- Finally, MVN assumption on variational posterior is limiting, in the sense that it is cannot captures multimodal distributions which the true posterior of NN exhibits. Implicit distributions have an advantage in this regards. Can we make this better?
  - We can maybe use approximate posterior in the form that being used by Louizos and Welling [2017] above. Is there a way to maintain tractability in this form? What if we use simple $p(\mathbf{z})$?

References

Arjun K Gupta and Daya K Nagar. *Matrix variate distributions*. Chapman and Hall/CRC, 1999.

Christos Louizos and Max Welling. Structured and efficient variational deep learning with matrix gaussian posteriors. In *International Conference on Machine Learning*, pages 1708–1716, 2016.

Shengyang Sun, Changyou Chen, and Lawrence Carin. Learning structured weight uncertainty in bayesian neural networks. In *Artificial Intelligence and Statistics*, pages 1283–1292, 2017.

Guodong Zhang, Shengyang Sun, David K. Duvenaud, and Roger B. Grosse. Noisy natural gradient as variational inference. *CoRR*, abs/1712.02390, 2017. URL http://arxiv.org/abs/1712.02390.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

David Ha, Andrew Dai, and Quoc V. Le. HyperNetworks. *arXiv:1609.09106 [cs]*, September 2016. URL http://arxiv.org/abs/1609.09106. arXiv: 1609.09106.

Nick Pawlowski, Andrew Brock, Matthew C. H. Lee, Martin Rajchl, and Ben Glocker. Implicit Weight Uncertainty in Neural Networks. *arXiv:1711.01297 [cs, stat]*, November 2017. URL http://arxiv.org/abs/1711.01297. arXiv: 1711.01297.

David Krueger, Chin-Wei Huang, Riashat Islam, Ryan Turner, Alexandre Lacoste, and Aaron Courville. Bayesian Hypernetworks. *arXiv:1710.04759 [cs, stat]*, October 2017. URL http://arxiv.org/abs/1710.04759. arXiv: 1710.04759.

Christos Louizos and Max Welling. Multiplicative normalizing flows for variational bayesian neural networks. *arXiv preprint arXiv:1703.01961*, 2017.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A Scalable Laplace Approximation for Neural Networks. February 2018. URL https://openreview.net/forum?id=Skdvd2xAZ.

Tim Salimans and Diederik P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 901–909, 2016.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417, 2015.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Networks. *arXiv:1505.05424 [cs, stat]*, May 2015. URL http://arxiv.org/abs/1505.05424. arXiv: 1505.05424.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

Thomas P Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc., 2001.

David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3): 448–472, 1992.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples (2014). *arXiv preprint arXiv:1412.6572*, 2014.

Diederik P Kingma, Tim Salimans, and Max Welling. Variational Dropout and the Local Reparameterization Trick. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2575–2583. Curran Associates, Inc., 2015. URL http://papers.nips.cc/paper/5666-variational-dropout-and-the-local-reparameterization-trick.pdf.