Learnable Bernoulli Dropout for Bayesian Deep Learning

Shahin Boluki[†] **Randy Ardywibowo**[†] **Siamak Zamani Dadaneh**[†]
Mingyuan Zhou[‡] **Xiaoning Qian**[†]
[†] Texas A&M University [‡]The University of Texas at Austin

Abstract

In this work, we propose learnable Bernoulli dropout (LBD), a new model-agnostic dropout scheme that considers the dropout rates as parameters jointly optimized with other model parameters. By probabilistic modeling of Bernoulli dropout, our method enables more robust prediction and uncertainty quantification in deep models. Especially, when combined with variational auto-encoders (VAEs), LBD enables flexible semi-implicit posterior representations, leading to new semi-implicit VAE (SIVAE) models. We solve the optimization for training with respect to the dropout parameters using Augment-REINFORCE-Merge (ARM), an unbiased and low-variance gradient estimator. Our experiments on a range of tasks show the superior performance of our approach compared with other commonly used dropout schemes. Overall, LBD leads to improved accuracy and uncertainty estimates in image classification and semantic segmentation. Moreover, using SIVAE, we can achieve state-of-the-art performance on collaborative filtering for implicit feedback on several public datasets.

1 INTRODUCTION

Deep neural networks (DNNs) are a flexible family of models that usually contain millions of free parameters. Growing concerns on overfitting of DNNs (Szegedy et al., 2013; Nguyen et al., 2015; Zhang et al., 2016; Bozhinoski et al., 2019) arise especially when considering their robustness and generalizability in real-world safety-critical applications such as autonomous driving and healthcare. To address this, Bayesian methods attempt to principally regularize and estimate the prediction uncertainty of DNNs. They introduce model uncertainty by placing prior distributions on the weights and biases of the networks. Since exact Bayesian inference is computationally intractable, many approximation methods have been developed such as Laplace approximation (MacKay, 1992a), Markov chain Monte Carlo (MCMC) (Neal, 2012), stochastic gradient MCMC (Welling and Teh, 2011; Ma et al., 2015; Springenberg et al., 2016), and variational inference methods (Blei et al., 2017; Hoffman et al., 2013; Blundell et al., 2015; Graves, 2011). In practice, these methods are significantly slower to train compared to non-Bayesian methods for DNNs, such as calibrated regression (Kuleshov et al., 2018), deep ensemble methods (Lakshminarayanan et al., 2017), and more recent prior networks (Malinin and Gales, 2018), which have their own limitations including training instability (Blum et al., 2019).

Although dropout, a commonly used technique to alleviate overfitting in DNNs, was initially used as a regularization technique during training (Hinton et al., 2012), Gal and Ghahramani (2016b) showed that when used at test time, it enables uncertainty quantification with Bayesian interpretation of the network outputs as Monte Carlo samples of its predictive distribution. Considering the original dropout scheme as multiplying the output of each neuron by a binary mask drawn from a Bernoulli distribution, several dropout variations with other distributions for random multiplicative masks have been studied, including Gaussian dropout (Kingma et al., 2015; Srivastava et al., 2014). Among them, Bernoulli dropout and extensions are most commonly used in practice due to their ease of implementation in existing deep architectures and their computation speed. Its simplicity and computational tractability has made Bernoulli dropout the current most popular method to introduce uncertainty in DNNs.

It has been shown that both the level of prediction accuracy and quality of uncertainty estimation are dependent on the network weight configuration as well

Proceedings of the 23[rd]International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

as the dropout rate (Gal, 2016). Traditional dropout mechanism with fixed dropout rate may limit model expressiveness or require tedious hand-tuning. Allowing the dropout rate to be estimated along with the other network parameters increases model flexibility and enables feature sparsity patterns to be identified. An early approach to learning dropout rates overlays a binary belief network on top of DNNs to determine dropout rates (Ba and Frey, 2013). Unfortunately, this approach does not scale well due to the significant model complexity increase. In (Zhuo et al., 2015) dropout rates for input features for linear classifiers are tuned by a hierarchical Bayesian approach.

Other dropout formulations instead attempt to replace the Bernoulli dropout with a different distribution. Following the variational interpretation of Gaussian dropout, Kingma et al. (2015) proposed to optimize the variance of the Gaussian distributions used for the multiplicative masks. However, in practice, optimization of the Gaussian variance is difficult. For example, the variance should be capped at 1 in order to prevent the optimization from diverging. This assumption limits the dropout rate to at most 0.5, and is not suitable for regularizing architectures with potentially redundant features, which should be dropped at higher rates. Also, Hron et al. (2017) showed that approximate Bayesian inference of Gaussian dropout is ill-posed since the improper log-uniform prior adopted in (Kingma et al., 2015) does not usually result in a proper posterior. Recently, a relaxed Concrete (Maddison et al., 2016) (Gumbell-Softmax (Jang et al., 2016)) distribution has been adopted in (Gal et al., 2017) to replace the Bernoulli mask for learnable dropout rate (Gal, 2016). However, the continuous relaxation introduces bias to the gradients which reduces its performance.

Motivated by recent efforts on gradient estimates for optimization with binary (discrete) variables (Yin and Zhou, 2019; Tucker et al., 2017; Grathwohl et al., 2017), we propose a learnable Bernoulli dropout (LBD) module for general DNNs. In LBD, the dropout probabilities are considered as variational parameters jointly optimized with the other parameters of the model. We emphasize that LBD exactly optimizes the true Bernoulli distribution of regular dropout, instead of replacing it by another distribution such as Concrete or Gaussian. LBD accomplishes this by taking advantage of a recent unbiased low-variance gradient estimator—Augment-REINFORCE-Merge (ARM) (Yin and Zhou, 2019)—to optimize the loss function of the deep neural network with respect to the dropout layer. This allows us to backpropagate through the binary random masks and compute unbiased, low-variance gradients with respect to the dropout parameters. This approach properly introduces learnable feature sparsity regularization to the deep network, improving the performance of deep architectures that rely on it. Moreover, our formulation allows each neuron to have its own learned dropout probability. We provide an interpretation of LBD as a more flexible variational Bayesian approximation method for learning Bayesian DNNs compared to Monte Carlo (MC) dropout. We combine this learnable dropout module with variational autoencoders (VAEs) (Kingma and Welling, 2013; Rezende et al., 2014), which naturally leads to a flexible semi-implicit variational inference framework with VAEs (SIVAE). Our experiments show that LBD results in improved accuracy and uncertainty quantification in DNNs for image classification and semantic segmentation compared with regular dropout, MC dropout (Gal and Ghahramani, 2016b), Gaussian dropout (Kingma et al., 2015), and Concrete dropout (Gal et al., 2017). More importantly, by performing optimization of the dropout rates in SIVAE, we achieve state-of-the-art performance in multiple different collaborative filtering benchmarks.

2 METHODOLOGY

2.1 Learnable Bernoulli Dropout (LBD)

Given a training dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, where $\boldsymbol{x}$ and y denote the input and target of interest respectively, a neural network is a function $f(\boldsymbol{x}; \boldsymbol{\theta})$ from the input space to the target space with parameters $\boldsymbol{\theta}$. The parameters are learned by minimizing an objective function $\mathcal{L}$, which is usually comprised of an empirical loss $\mathcal{E}$ with possibly additional regularization terms $\mathcal{R}$, by stochastic gradient descent (SGD):

$$\mathcal{L}(\boldsymbol{\theta}|\mathcal{D}) \approx \frac{N}{M} \sum_{i=1}^{M} \mathcal{E}(f(\boldsymbol{x}_i; \boldsymbol{\theta}), y_i) + \mathcal{R}(\boldsymbol{\theta}), \qquad (1)$$

where M is the mini-batch size.

Consider a neural network with L fully connected layers. The j^{th} fully connected layer with K_j neurons takes the output of the $(j-1)^{th}$ layer with K_{j-1} neurons as input. We denote the weight matrix connecting layer $j-1$ to j by $W_j \in \mathbb{R}^{K_{j-1} \times K_j}$. Dropout takes the output to each layer and multiplies it with a random variable $\boldsymbol{z}_j \sim p(\boldsymbol{z}_j)$ element-wise (channel-wise for convolutional layers). The most common choice for $p(\boldsymbol{z}_j)$ is the Bernoulli distribution $\text{Ber}(\sigma(\alpha_j))$ with dropout rate $1 - \sigma(\alpha_j)$, where we have reparameterized the dropout rate using the sigmoid function $\sigma(\cdot)$. With this notation, let $\boldsymbol{\alpha} = \{\alpha_j\}_{j=1}^L$ denote the collection of all logits of the dropout parameters, and let $\boldsymbol{z} = \{\boldsymbol{z}_j\}_{j=1}^L$ denote the collection of all dropout masks. Dropout in this form is one of the most common regularization techniques in training DNNs to avoid

overfitting and improve generalization and accuracy on unseen data. This can also be considered as using a data dependent weight; if $z_{jk} = 0$ for input $\boldsymbol{x}$, then the kth row of W_j will be set to zero.

The parameter $\boldsymbol{\alpha}$ of the random masks $\boldsymbol{z}$ has been mainly treated as hyperparameters in the literature, requiring tuning by grid-search, which is prohibitively expensive. Instead, we propose to learn the dropout rates for Bernoulli masks jointly with the other model parameters. Specifically, we aim to optimize the expectation of the loss function with respect to the Bernoulli distributed dropouts:

$$\min_{\boldsymbol{\theta}=\{\boldsymbol{\theta}\setminus\boldsymbol{\alpha},\boldsymbol{\alpha}\}} \quad \mathbb{E}_{\boldsymbol{z}\sim\prod_{i=1}^{M}\mathrm{Ber}(\boldsymbol{z}_i;\sigma(\boldsymbol{\alpha}))}\big[\mathcal{L}(\boldsymbol{\theta},\boldsymbol{z}|\mathcal{D})\big]. \tag{2}$$

We next formulate the problem of learning dropout rates for supervised feed-forward DNNs, and unsupervised latent representation learning in VAEs. In the formulations that follow, the dropout rates can be optimized using the method described in Section 2.4. We first briefly review the variational interpretation of dropout in Bayesian deep neural networks and show how our LBD fits here. Then, we discuss how the dropout rates can be adaptive to the data in the context of VAEs. Specifically, combining the Bernoulli dropout layer with VAEs allows us to construct a semi-implicit variational inference framework.

2.2 Variational Bayesian Inference with LBD

In Bayesian neural networks (BNNs) (MacKay, 1992b; Neal, 1995), instead of finding point estimates of the weight matrices, the goal is to learn a distribution over them. In this setup, a prior is assumed over the weight matrices, $p(W)$, which is updated to a posterior given the training data following Bayes' rule $p(W|\mathcal{D}) = \frac{p(\mathcal{D}|W)p(W)}{p(\mathcal{D})}$. This posterior distribution captures the set of plausible models and imposes a predictive distribution for the target of a new data point. Due to the intractability of calculating $p(\mathcal{D})$, different approximation techniques have been developed (Blei et al., 2017; Graves, 2011; Blundell et al., 2015; Gal and Ghahramani, 2016b), which use a simple (variational) distribution $q_{\boldsymbol{\theta}}(W)$ to approximate the posterior. By taking this approach the intractable marginalization in the original inference problem is replaced by an optimization problem, where the parameters $\boldsymbol{\theta}$ are optimized by fitting $q_{\boldsymbol{\theta}}(W)$ to $p(W|\mathcal{D})$. Following the variational interpretation of Bernoulli dropout, the approximating distribution is a mixture of two Gaussian distributions with very small variance, where one mixture component has mean zero (Gal and Ghahramani, 2016b,a). Assuming a network with L layers, we denote the collection of all weight matrices by $\boldsymbol{W} = \{W_j\}_{j=1}^{L}$.

Under this formulation, we propose learnable Bernoulli dropout (LBD) as a variational approximation. Unlike the common assumption where all neurons in layer j share the same dropout rate, in our approach, we let each neuron k in each layer have its own dropout rate α_{jk}. Thus, each layer has a mean weight matrix M_j and dropout parameters $\boldsymbol{\alpha}_j = \{\alpha_{jk}\}_{k=1}^{K_{j-1}}$. With this, our variational distribution consists of the parameters $\boldsymbol{\theta} = \{M_j, \boldsymbol{\alpha}_j\}_{j=1}^{L}$. In this setup, $q_{\boldsymbol{\theta}}(\boldsymbol{W}) = \prod_{j=1}^{L} q_{\boldsymbol{\theta}}(W_j)$, where $q_{\boldsymbol{\theta}}(W_j) = M_j^T \mathrm{diag}(\mathrm{Ber}(\boldsymbol{\alpha}_j))$, and the objective function for optimizing the variational parameters is

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta} = \{M_j, \boldsymbol{\alpha}_j\}_{j=1}^{L}|\mathcal{D}) = -\frac{N}{M}\sum_{i=1}^{M}\log p(y_i|f(\boldsymbol{x}_i;\boldsymbol{W}_i)) \\ + \mathrm{KL}(q_{\boldsymbol{\theta}}(\boldsymbol{W})||p(\boldsymbol{W})),\end{aligned} \tag{3}$$

where $\boldsymbol{W}_i$ denotes the realization of the random weight matrices for each data point. The likelihood function $p(y_i|f(\boldsymbol{x}_i;\boldsymbol{W}_i))$ is usually a softmax or a Gaussian for classification and regression problems, respectively. The Kullback-Leibler (KL) divergence term is a regularization term that prevents the approximate posterior from deviating too far from the prior. By employing the quantized zero-mean Gaussian prior in (Gal, 2016) with variance s^2, we have $\mathrm{KL}(q_{\boldsymbol{\theta}}(\boldsymbol{W})||p(\boldsymbol{W})) \propto \sum_{j=1}^{L}\sum_{k=1}^{K_{j-1}} \frac{\alpha_{jk}}{2s^2}||M_j[\cdot,k]||^2 - \mathcal{H}(\alpha_{jk})$, where $M_j[\cdot,k]$ represents the k^{th} column of the mean weight matrix M_j and $\mathcal{H}(\alpha_{jk})$ denotes the entropy of a Bernoulli random variable with parameter α_{jk}.

After fitting the approximate posterior distribution, the posterior predictive $p(y|\boldsymbol{x},\mathcal{D})$ for the target of a new data point $\boldsymbol{x}$ is approximated by Monte Carlo integration with S samples obtained by stochastic forward passes as $\frac{1}{S}\sum_{s=1}^{S} p(y|f(\boldsymbol{x};\boldsymbol{W}^{(s)})$. The entropy of the posterior predictive $p(y|\boldsymbol{x},\mathcal{D})$ can be considered as a measure of uncertainty (Mukhoti and Gal, 2018; Gal, 2016).

Note that the variational interpretation and the derivation of the objective function in this section corresponds to Bernoulli dropout, i.e. when the variational distributions are constructed by multiplications with Bernoulli random variables. Replacing the Bernoulli distribution with a relaxed distribution like Concrete introduces an additional level of approximation and bias which can lead to degraded predictive and uncertainty quantification performance.

2.3 Combining VAE and LBD into SIVAE

VAEs (Kingma and Welling, 2013; Rezende et al., 2014) have been widely used for amortized inference and unsupervised feature learning. They tie together the modeling and inference through an encoder-decoder

architecture. The encoder (data-dependent variational posterior) $q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x}_i)$ and decoder $p_{\boldsymbol{\omega}}(\boldsymbol{x}_i|\boldsymbol{\eta})$ (generative model) are based on neural networks parameterized by $\boldsymbol{\psi}$ and $\boldsymbol{\omega}$, respectively, which are inferred by minimizing the negative evidence lower bound (ELBO):

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}=\{\boldsymbol{\psi},\boldsymbol{\omega}\}|\mathcal{D}) = -\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x}_i)}\big[\log p_{\boldsymbol{\omega}}(\boldsymbol{x}_i|\boldsymbol{\eta})\big] \\ + \quad \beta\mathrm{KL}(q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x}_i)||p(\boldsymbol{\eta})),\end{aligned} \tag{4}$$

where the prior over the latent variables, $p(\boldsymbol{\eta})$, is commonly set to $\mathcal{N}(0, I)$. While $\beta = 1$ in the original VAE (Kingma and Welling, 2013), setting either $\beta > 1$ or $\beta < 1$ have been proposed (Higgins et al., 2017; Zhao et al., 2018) to allow for learning more informative latent representations or maximization of the mutual information between latent and observed variables. Regular dropout has been commonly used in training VAEs to prevent overfitting. When training, one sample is taken for each observed point in the mini-batch and the network parameters are found by minimizing (4). The dropout rate can be tuned by grid-search to find the highest ELBO. This approach is prohibitively expensive for large (deep/wide) models.

Instead of using Gaussian for the data-dependent variational distribution $q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x}) = \mathcal{N}(\mu_{\boldsymbol{\psi}}(\boldsymbol{x}), \mathrm{diag}(\Sigma_{\boldsymbol{\psi}}(\boldsymbol{x})))$, we adopt the semi-implicit variational inference (SIVI) approach (Yin and Zhou, 2018), allowing us to use more expressive data-dependent variational distributions. Here, the variational distribution consists of an explicit conditional distribution $q(\boldsymbol{\eta}|\boldsymbol{\gamma})$ mixed with an implicit distribution $q(\boldsymbol{\gamma})$. This allows the construction of a semi-implicit VAE (SIVAE), where the first stochastic layer, $q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x})$, is still a Gaussian distribution, but other stochastic layers are constructed implicitly in the upper layers of the encoder. In (Yin and Zhou, 2018), the stochastic layers were constructed by concatenating the deterministic layer output with random noise and observed data to form the input to the next layer.

In our approach, dropout can be considered to introduce an implicit mixing effect that constructs a SIVAE. More specifically, a realization of the local dropout mask results in a predicted mean and variance for the Gaussian variational distribution $\mathcal{N}(\mu_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z}), \mathrm{diag}(\Sigma_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z})))$. By marginalizing over the dropout, we can construct the following implicit variational distribution:

$$\begin{aligned}q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x}) =& \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{\alpha}}(\boldsymbol{z})}[q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x}, \boldsymbol{z})] \\ =& \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{\alpha}}(\boldsymbol{z})}[\mathcal{N}(\mu_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z}), \mathrm{diag}(\Sigma_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z})))],\end{aligned} \tag{5}$$

which is more flexible than the common Gaussian assumption and can result in better latent representations.

Using an asymptotically exact ELBO for SIVAE (Yin and Zhou, 2018), we infer both the encoder and decoder network parameters together with the dropout rates by optimizing

$$\begin{aligned}&\mathcal{L}(\boldsymbol{\theta}=\{\boldsymbol{\psi},\boldsymbol{\omega},\boldsymbol{\alpha}\}|\mathcal{D}) = \\ &-\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{\boldsymbol{z}_i\sim p_{\boldsymbol{\alpha}}(\boldsymbol{z}),\boldsymbol{\eta}\sim q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x}_i,\boldsymbol{z}_i),\boldsymbol{z}_i^{(1)},\cdots,\boldsymbol{z}_i^{(V)}\sim p_{\boldsymbol{\alpha}}(\boldsymbol{z})}\Big[\\ &\log p_{\boldsymbol{\omega}}(\boldsymbol{x}_i|\boldsymbol{\eta})p(\boldsymbol{\eta}) - \log\frac{1}{V+1}\big[q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x}_i,\boldsymbol{z}_i)+ \\ &\qquad\qquad\sum_{v=1}^{V}q_{\boldsymbol{\psi}}(\boldsymbol{\eta}|\boldsymbol{x}_i,\boldsymbol{z}_i^{(v)})\big]\Big],\end{aligned} \tag{6}$$

as discussed in the next section.

2.4 Optimization of LBD

The optimization of (2) with respect to global parameters $\boldsymbol{\theta}\backslash\boldsymbol{\alpha}$ can be performed by sampling a dropout mask for each data point in the mini-batch in the forward pass and calculating the gradients with respect to those parameters with SGD. The optimization with respect to dropout rates, however, is challenging as the reparameterization technique (a.k.a. path-wise derivative estimator) (Kingma and Welling, 2013; Rezende et al., 2014) cannot be used. On the other hand, score-function gradient estimators such as REINFORCE (Williams, 1992; Fu, 2006) possess high estimation variance.

In this paper, we estimate the gradient of (2) with respect to $\boldsymbol{\alpha}$ with the ARM estimator (Yin and Zhou, 2019). Using ARM, we are able to directly optimize the Bernoulli dropout rates without introducing any bias. For a vector of K binary random variables $\boldsymbol{z} = [z_1, \cdots, z_K]$ parameterized by $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_K]$, the logits of the Bernoulli probability parameters, the gradient of $\mathbb{E}_{\boldsymbol{z}}[\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{z})]$ with respect to $\boldsymbol{\alpha}$ can be expressed as (Yin and Zhou, 2019)

$$\begin{aligned}\nabla_{\boldsymbol{\alpha}}\mathbb{E}_{\boldsymbol{z}}[\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{z})] =& \mathbb{E}_{\boldsymbol{u}\sim\prod_{k=1}^{K}\mathrm{Unif}_{[0,1]}(u_k)}\Big[\big(\mathcal{L}(\boldsymbol{\theta}, 1_{[\boldsymbol{u}>\sigma(-\boldsymbol{\alpha})]}) \\ &- \mathcal{L}(\boldsymbol{\theta}, 1_{[\boldsymbol{u}<\sigma(\boldsymbol{\alpha})]})\big)\big(\boldsymbol{u}-\frac{1}{2}\big)\Big].\end{aligned} \tag{7}$$

Here Unif denotes the uniform distribution, and $\mathcal{L}(\boldsymbol{\theta}, 1_{[\boldsymbol{u}<\sigma(\boldsymbol{\alpha})]})$ denotes the loss obtained by setting $\boldsymbol{z} = 1_{[\boldsymbol{u}<\sigma(\boldsymbol{\alpha})]} := \big(1_{[u_1<\sigma(\alpha_1)]}, \cdots, 1_{[u_K<\sigma(\alpha_K)]}\big)$. With this gradient estimate, we can proceed to compute the gradient of our loss function in (1), with special cases in (3) and (6). Note that the regularization term in (1) is usually not a function of $\boldsymbol{z}$. For example, the KL divergence term in many Bayesian and Bayesian deep learning formulations (e.g. the one in Section 2.2) only depends on the distribution of the model parameters.

With this, the gradient of the objective function with respect to the Bernoulli dropout parameters can be expressed as

$$\nabla_{\boldsymbol{\alpha}}\mathbb{E}_{\boldsymbol{z}}\big[\mathcal{L}(\boldsymbol{\theta},\boldsymbol{z}|\mathcal{D})\big] = \frac{N}{M}\sum_{i=1}^{M}\mathbb{E}_{\boldsymbol{u}_i\sim\prod_{k=1}^{K}\mathrm{Unif}_{[0,1]}(u_{ik})}\Big[\big(\mathcal{E}(1_{[\boldsymbol{u}_i>\sigma(-\boldsymbol{\alpha})]})- \mathcal{E}(1_{[\boldsymbol{u}_i<\sigma(\boldsymbol{\alpha})]})\big)\big(\boldsymbol{u}_i-\frac{1}{2}\big)\Big] + \nabla_{\boldsymbol{\alpha}}\mathcal{R}(\boldsymbol{\alpha}). \quad (8)$$

Here, $\mathcal{E}(1_{[\boldsymbol{u}_i>\sigma(-\boldsymbol{\alpha})]})$ is the empirical loss obtained by setting $\boldsymbol{z}$ to 1 if $\boldsymbol{u}_i > \sigma(-\boldsymbol{\alpha})$, and similarly for $\mathcal{E}(1_{[\boldsymbol{u}_i<\sigma(\boldsymbol{\alpha})]})$. Note that the expectation can be estimated using only one sample, allowing us to estimate the gradient efficiently.

3 RESULTS AND DISCUSSION

We evaluate LBD on three different tasks: image classification, semantic segmentation, and collaborative filtering. Before presenting these results, we investigate the performance on a toy example, where we can calculate the true gradients exactly, to demonstrate the advantages of LBD over the existing dropout schema. We implement all of our methods in Tensorflow (Abadi et al., 2016) on a single cluster node with Intel Xeon E5-2680 v4 2.40 GHz processor and a Tesla K80 Accelerator.

3.1 Toy Example

In this section, we investigate the bias and bias-variance trade-off of gradient estimates for dropout parameters from LBD and Concrete with respect to a simple Bernoulli dropout model where exact calculation of the gradients is tractable. For this purpose, we consider a toy regression task with simulated data. The base model for our example is a simple neural network with one input, one output and two hidden layer nodes, with ReLU non-linearity and dropout applied after hidden layer neurons. There are two dropout parameters, α_1 and α_2 for the hidden layer. The objective function is $\mathbb{E}_{\boldsymbol{z}_i\sim\prod_{i=1}^{N}\mathrm{Ber}(\boldsymbol{z}_i;\sigma([\alpha_1,\alpha_2]))}\big[(y_i - f(x_i;W,z_i))^2\big]$. The data (3000 samples) is generated from the same model without the non-linearity and dropout (i.e. a linear model). The weights of the network are randomly initialized and held fixed during training. We calculate the true gradients for dropout parameters at different values of $\sigma(\alpha_1)$ and $\sigma(\alpha_2)$ and compare them with estimates by LBD and Concrete in the histograms of Figure 1. The bias, standard deviation (STD), and mean squared error (MSE) of the estimates for α_1 from LBD and Concrete calculated by 200 Monte Carlo samples are shown in panels (a)-(f), respectively, while additional results for α_2 are included in the Supplementary. LBD which leverages ARM clearly has no bias and lower MSE in estimating the gradients w.r.t. the Bernoulli model. We also provide trace plots of the estimated gradients by LBD and Concrete using 50 Monte Carlo samples when updating the parameters via gradient descent with the true gradients in the bottom panels in Figure 1. Panels (g) and (h) correspond to α_1 and α_2, respectively. The estimates from LBD follow the true gradients very closely.

3.2 Image Classification and Semantic Segmentation

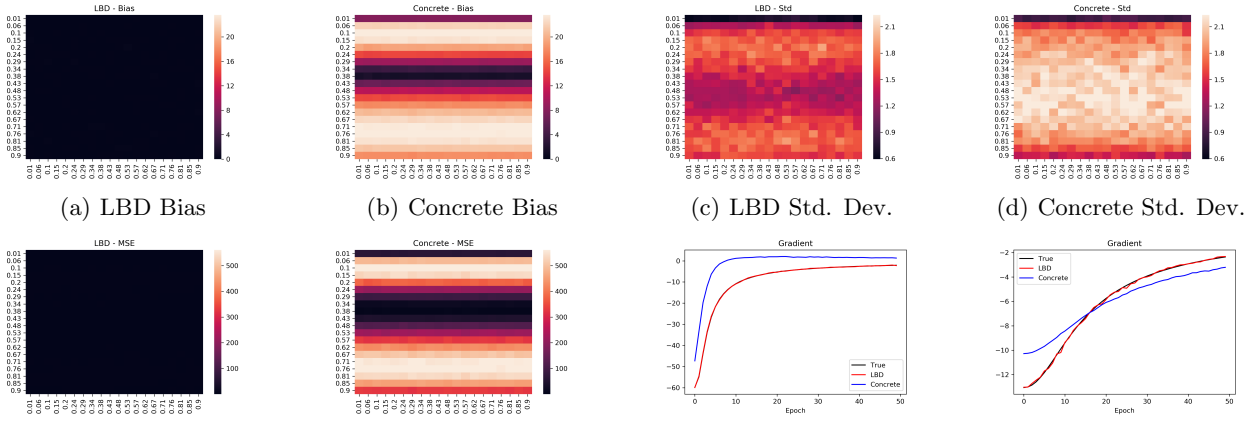
In image classification and segmentation, we evaluate different dropout results based on prediction accuracy and mean Intersection over Union (IoU) respectively. To assess the quality of uncertainty estimation, we use the PAvPU (P(Accurate|Certain) vs P(Uncertain|Inaccurate)) uncertainty evaluation metric proposed in (Mukhoti and Gal, 2018). P(Accurate|Certain) is the probability that the model is accurate on its output given that it is confident on the same, while P(Uncertain|Inaccurate) is the probability that the model is uncertain about its output given that it has made a mistake in prediction. Combining these together results in the PAvPU metric, calculated by the following equation:

$$\mathrm{PAvPU} = \frac{n_{iu} + n_{ac}}{n_{ac} + n_{au} + n_{iu} + n_{ic}}. \quad (9)$$

Here, n_{ac} is the number of accurate and certain predictions; n_{ic} is the number of inaccurate and certain predictions; n_{au} is the number of accurate and uncertain predictions; and n_{iu} is the number of inaccurate and uncertain predictions. Intuitively, this metric gives a high score to models that predict accurately with confidence or put high uncertainty estimates for incorrect predictions. To determine when the model is certain or uncertain, the mean predictive entropy can be used as a threshold. Or alternatively, different thresholds of the predictive entropy can be used: e.g. denoting the minimum and maximum predictive entropy for the validation dataset as $\mathrm{PredEnt}_{\min}$ and $\mathrm{PredEnt}_{\max}$, different thresholds $\mathrm{PredEnt}_{\min} + t(\mathrm{PredEnt}_{\max} - \mathrm{PredEnt}_{\min})$ can be considered by fixing $t \in [0,1]$. In other words, we would classify a prediction as uncertain if its predictive entropy is greater than a certain threshold.

3.2.1 Image classification on CIFAR-10

We evaluate LBD under different configurations on the CIFAR-10 classification task (Krizhevsky, 2009), and compare its accuracy and uncertainty quantification with other dropout configurations on the VGG19 architecture (Simonyan and Zisserman, 2014). The



(a) LBD Bias (b) Concrete Bias (c) LBD Std. Dev. (d) Concrete Std. Dev.

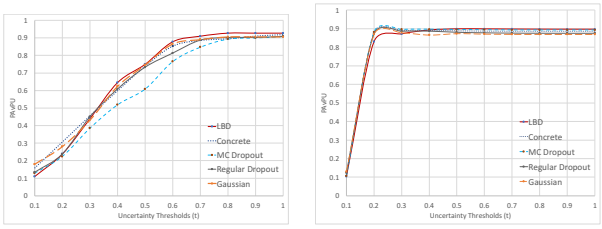
(e) LBD MSE (f) Concrete MSE (g) 1st Param. Gradient (h) 2nd Param. Gradient

Figure 1: Comparison of gradient estimates for dropout parameters in the toy example.

CIFAR-10 dataset contains 50,000 training images and 10,000 testing images of size 32×32 from 10 different classes. The VGG architecture consists of two layers of dropout before the output. Each layer of dropout is preceded by a fully connected layer followed by a Rectified Linear Unit (ReLU) nonlinearity. We modify the existing dropout layers in this architecture using LBD as well as other forms of dropout. For all experiments, we utilize a batch size of 64 and train for 300 epochs using the Adam optimizer with a learning rate of 1×10^{-5} (Kingma and Ba, 2014). We use weights pretrained on ImageNet, and resize the images to 224×224 to fit the VGG19 architecture. No other data augmentation or preprocessing is done.

We compare our LBD with regular dropout, MC dropout (Gal and Ghahramani, 2016b), and concrete dropout (Gal et al., 2017). For regular and MC dropout, we utilize the hand-tuned dropout rate of 0.5. To obtain predictions and predictive uncertainty estimates, we sample 10 forward passes of the architecture and calculate (posterior) mean and predictive entropy. The prediction accuracy and uncertainty quantification results for the last epoch of training is shown in Figure 2(a) and Table 1. Our LBD consistently achieves better accuracy and uncertainty estimates compared to other methods that learn or hand-tune the dropout rate.

We further test LBD with a shared parameter for all dropout layers. The accuracy achieved in this case is 92.11%, which is higher than the cases with hand-tuned dropout rates, but lower than LBD with different dropout rates for different layers included in Table 1. This shows that even optimizing one dropout rate can still be beneficial compared to considering it as a hyperparameter and hand-tuning it. Moreover, it further confirms the need for the model adaptability achieved by learning more flexible dropout rates from data.



(a) CIFAR-10 (b) CamVid

Figure 2: PAvPU for different models under different uncertainty thresholds for classification on CIFAR-10 and segmentation for CamVid

3.2.2 Semantic segmentation on CamVid

We evaluate our LBD for image segmentation on the CamVid dataset (Brostow et al., 2009). This contains 367 training images, 101 validation images, and 232 testing images of size 480×360 with 11 semantic segmentation classes. In all experiments, we train on the training images and test on the testing images. We use the FC-DenseNet-103 architecture as the base model (Jégou et al., 2017). This architecture consists of 11 blocks, with the middle block having 15 layers. Each layer consists of a ReLU nonlinearity followed by a convolution and dropout. Due to the concatenation of many features within each resolution, dropout becomes important to regularize the network by removing less relevant features from consideration. In our experiments, we replace the dropout in the 15 middle layers of this architecture with LBD as well as other forms of dropout. Here, the dropout parameters are shared between neurons in the same layer. For all experiments, we utilize a batch size of 1 and train for 700 epochs. We did not crop the 480×360 sized image and performed horizontal image flipping for data augmentation. We train our model using the RMSProp optimizer using a decay of 0.995 (Hinton et al.) and a learning rate of

Table 1: Comparison of LBD accuracy and uncertainty quantification with other forms of dropout for classification on the CIFAR-10 dataset and semantic segmentation on the CamVid dataset. (All numbers in %)

	CIFAR-10		CamVid		
Method	Accuracy	Mean PAvPU	Accuracy	Mean IoU	Mean PAvPU
LBD	**93.47**	**64.52**	**89.18**	**60.42**	**85.16**
Concrete	92.72	59.97	88.64	55.95	83.81
Gaussian	91.16	52.71	88.04	54.85	85.02
MC Dropout	91.57	52.08	87.80	54.40	84.39
Regular Dropout	91.61	56.81	86.65	53.62	81.17

0.0001. For regular and MC dropout, we utilize the hand-tuned dropout rate of 0.2.

For uncertainty estimates in image segmentation, each pixel can be individually classified into certain or uncertain; however, Mukhoti and Gal (2018) noted that capturing uncertainties occurring in regions comprising of multiple pixels in close neighborhoods would be more useful for downstream machine learning tasks. Thus, they opted to compute patch-wise uncertainties by averaging the entropy over a sliding window of size $w \times w$. In our evaluation, we use a window of size 2×2. The results of our semantic segmentation experiments are shown in Table 1 and Figure 2(b). As shown in the table and figure, our LBD consistently achieves better accuracy and uncertainty estimates, compared to other methods that either learn or hand-tune the dropout rates. It is also interesting to note that methods that are based on Bayesian approximation which learn the dropout parameters generally had better PAvPU than MC and regular dropout, confirming that Bayesian deep learning methods provide better uncertainty estimates when using optimized dropout rates.

3.3 Collaborative Filtering for Implicit Feedback Data

Collaborative filtering which predicts user preferences by discovering similarity patterns among users and items (Herlocker et al., 2004) is among the most notable algorithms for recommender systems. VAEs with multinomial likelihood have been shown to provide state-of-the-art performance for collaborative filtering on implicit feedback data (Liang et al., 2018). They are especially interesting for large-scale datasets due to the amortized inference.

Our experimental setup is similar to (Liang et al., 2018). Following their heuristic search for β in the VAE training objective, we also gradually increase β from 0 to 1 during training and record the β that maximizes the validation performance. For all variations of VAE and our SIVAE we use the multinomial likelihood. For all encoder and decoder networks the dimension of the latent representations is set to 200, with one hidden layer of 600 neurons.

The experiments are performed on three user-item datasets: MovieLens-20M (ML-20M) (Harper and Konstan, 2016), Netflix Prize (Netflix) (Bennett et al., 2007), and Million Song Dataset (MSD) (Bertin-Mahieux et al., 2011). We take similar pre-processing steps as in (Liang et al., 2018). For ML-20M and Netflix, users who have watched less than 5 movies are discarded and user-movie matrix is binarized by keeping the ratings of 4 and higher. For MSD, users with at least 20 songs in their listening history and songs that are listened to by at least 200 users are kept and the play counts are binarized. After pre-processing, the ML-20M dataset contains around 136,000 users and 20,000 movies with 10M interactions; Netflix contains around 463,000 user and 17,800 item with 56.9M interactions; MSD has around 571,000 user and 41,000 song with 33.6M interactions left.

To evaluate the performance of different models, two of the most popular learning-to-rank scoring metrics are employed, Recall@R and the truncated normalized discounted cumulative gain (NDCG@R). Recall@R treats all items ranked within the first R as equally important while NDCG@R considers a monotonically increasing discount coefficient to emphasize the significance of higher ranks versus lower ones. Following (Liang et al., 2018), we split the users into train/validation/test sets. All the interactions of users in training set are used during training. For validation and test sets, 80% of interactions are randomly selected to learn the latent user-level representations, with the other 20% held-out to calculate the evaluation metrics for model predictions. For ML-20M, Netflix, and MSD datasets 10,000, 40,000, and 50,000 users are assigned to each of the validation and test sets, respectively.

We compare the performance of VAE (with dropout at input layers) and DAE (Liang et al., 2018) (VAE+Drop and DAE) with our proposed SIVAE with learnable Bernoulli dropout (SIVAE+LBDrop). We also provide the results for SIVAE with learnable Concrete dropout (SIVAE+CDrop) and SIVAE with learnable

Table 2: Comparisons of various baselines and different configurations of VAE and dropout with multinomial likelihood on ML-20M, Netflix and MSD dataset. The standard errors are around 0.2% for ML-20M and 0.1% for Netflix and MSD. (All numbers in %)

Method	ML-20M/Netflix/MSD		
	Recall@20	Recall@50	NDCG@100
SIVAE+LBDrop	**39.95/35.63/27.18**	**53.95/44.70/37.36**	**43.01/39.04/32.33**
SIVAE+GDrop	35.77/31.43/22.50	49.45/40.68/31.01	38.69/35.05/26.99
SIVAE+CDrop	37.19/32.07/24.32	51.30/41.52/33.19	39.90/35.67/29.10
VAE-VampPrior+Drop	39.65/35.15/26.26	53.63/44.43/35.89	42.56/38.68/31.37
VAE-VampPrior	35.84/31.09/22.02	50.27/40.81/30.33	38.57/34.86/26.57
VAE-IAF+Drop	39.29/34.65/25.65	53.52/44.02/34.96	42.37/38.21/30.67
VAE-IAF	35.94/32.77/21.92	50.28/41.95/30.23	38.85/36.23/26.50
VAE+Drop	39.47/35.16/26.36	53.53/44.38/36.21	42.63/38.60/31.33
VAE	35.67/31.12/21.98	49.89/40.78/30.33	38.53/34.86/26.49
DAE	38.58/34.50/25.89	52.28/43.41/35.23	41.92/37.85/31.04
WMF	36.00/31.60/21.10	49.80/40.40/31.20	38.60/35.10/25.70
SLIM	37.00/34.70/-	49.50/42.80/-	40.10/37.90/-
CDAE	39.10/34.30/18.80	52.30/42.80/28.30	41.80/37.60/23.70

Gaussian dropout (SIVAE+GDrop), where we have replaced the learnable Bernoulli dropout in our SIVAE construction of Section 2.3 with Concrete and Gaussian versions. Furthermore, the results for VAE without any dropout (VAE) are shown in Table 2. Note that because of the discrete (binary) nature of the implicit feedback data, Bernoulli dropout seems a naturally better fit for the input layers compared with other relaxed distributions due to not changing the nature of the data. We also test two extensions of VAE-based models, VAE with variational mixture of posteriors (VAE-VampPrior) (Tomczak and Welling, 2018) and VAE with inverse autoregressive flow (VAE-IAF) (Kingma et al., 2016) with and without dropout layer for this task. The prior in VAE-VampPrior consists of mixture distributions with components given by variational posteriors conditioned on learnable pseudo-inputs, where the encoder parameters are shared between the prior and the variaional posterior. VAE-IAF uses invertible transformations based on an autoregressive neural network to build flexible variational posterior distributions. We train all models with Adam optimizer (Kingma and Ba, 2014). The models are trained for 200 epochs on ML-20M, and for 100 epochs on Netflix and MSD. For comparison purposes, we have included the published results for weighted matrix factorization (WMF) (Hu et al., 2008), SLIM (Ning and Karypis, 2011), and collaborative denoising autoencoder (CDAE) (Wu et al., 2016) on these datasets. WMF is a linear low-rank factorization model which is trained by alternating least squares. SLIM is a sparse linear model which learns an item-to-item similarity matrix by solving a constrained ℓ_1-regularization optimization. CDAE augments the standard DAE by adding per-user latent factors to the input.

From the table, we can see that the SIVAE with learnable Bernoulli dropout achieves the best performance in all metrics on all datasets. Note that dropout is crucial to VAE's good performance and removing it has a huge impact on its performance. Also, Concrete or Gaussian dropout which also attempt to optimize the dropout rates performed worse compared to VAE with regular dropout.

4 CONCLUSION

In this paper we have proposed learnable Bernoulli dropout (LBD) for model-agnostic dropout in DNNs. Our LBD module improves the performance of regular dropout by learning adaptive and per-neuron dropout rates. We have formulated the problem for non-Bayesian and Bayesian feed-forward supervised frameworks as well as the unsupervised setup of variational auto-encoders. Although Bernoulli dropout is universally used by many architectures due to their ease of implementation and faster computation, previous approaches to automatically learning the dropout rates from data have all involved replacing the Bernoulli distribution with a continuous relaxation. Here, we optimize the Bernoulli dropout rates directly using the Augment-REINFORCE-Merge (ARM) algorithm. Our experiments on computer vision tasks demonstrate that for the same base models, adopting LBD results in better accuracy and uncertainty quantification compared with other dropout schemes. Moreover, we have shown that combining LBD with VAEs, naturally leads to a flexible semi-implicit variational inference framework with VAEs (SIVAE). By optimizing the dropout rates in SIVAE, we have achieved state-of-the-art performance in multiple collaborative filtering benchmarks.

Acknowledgement

The presented materials are based upon the work supported by the National Science Foundation under Grants CCF-1553281, IIS-1812641, IIS-1812699, and CCF-1934904. We also thank Texas A&M High Performance Research Computing and Texas Advanced Computing Center for providing computational resources to perform experiments in this work.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, OSDI'16, pages 265–283, Berkeley, CA, USA, 2016. USENIX Association. ISBN 978-1-931971-33-1.

Jimmy Ba and Brendan Frey. Adaptive dropout for training deep neural networks. In *Advances in Neural Information Processing Systems*, pages 3084–3092, 2013.

James Bennett, Stan Lanning, et al. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, NY, USA., 2007.

Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere. The million song dataset. 2011.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112 (518):859–877, 2017.

Hermann Blum, Paul-Edouard Sarlin, Juan Nieto, Roland Siegwart, and Cesar Cadena. The fishyscapes benchmark: Measuring blind spots in semantic segmentation. *arXiv preprint arXiv:1904.03215*, 2019.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

Darko Bozhinoski, Davide Di Ruscio, Ivano Malavolta, Patrizio Pelliccione, and Ivica Crnkovic. Safety for mobile robotic systems: A systematic mapping study from a software engineering perspective. *Journal of Systems and Software*, 151:150–179, 2019.

Gabriel J Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2):88–97, 2009.

Michael C Fu. Gradient estimation. *Handbooks in operations research and management science*, 13:575–616, 2006.

Yarin Gal. *Uncertainty in deep learning*. PhD thesis, University of Cambridge, 2016.

Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with Bernoulli approximate variational inference. In *ICLR workshop track*, 2016a.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016b.

Yarin Gal, Jiri Hron, and Alex Kendall. Concrete dropout. In *Advances in Neural Information Processing Systems*, pages 3581–3590, 2017.

Will Grathwohl, Dami Choi, Yuhuai Wu, Geoffrey Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. *arXiv preprint arXiv:1711.00123*, 2017.

Alex Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356, 2011.

F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):19, 2016.

Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1): 1303–1347, 2013.

Jiri Hron, Alexander G de G Matthews, and Zoubin Ghahramani. Variational gaussian dropout is not bayesian. *arXiv preprint arXiv:1711.02989*, 2017.

Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272. IEEE, 2008.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-softmax. In *International Conference on Learning Representations*, 2016.

Simon Jégou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 11–19, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pages 2575–2583, 2015.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in neural information processing systems*, pages 4743–4751, 2016.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. *arXiv preprint arXiv:1807.00263*, 2018.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 689–698. International World Wide Web Conferences Steering Committee, 2018.

Y. Ma, T. Chen, and E. Fox. A complete recipe for stochastic gradient MCMC. In *NIPS*, pages 2899–2907, 2015.

David JC MacKay. *Bayesian methods for adaptive models*. PhD thesis, California Institute of Technology, 1992a.

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

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The Concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2016.

Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems*, pages 7047–7058, 2018.

Jishnu Mukhoti and Yarin Gal. Evaluating bayesian deep learning methods for semantic segmentation. *arXiv preprint arXiv:1811.12709*, 2018.

Radford M Neal. *Bayesian learning for neural networks*. PhD thesis, University of Toronto, 1995.

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

Xia Ning and George Karypis. Slim: Sparse linear methods for top-n recommender systems. In *2011 IEEE 11th International Conference on Data Mining*, pages 497–506. IEEE, 2011.

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

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Jost Tobias Springenberg, Aaron Klein, Stefan Falkner, and Frank Hutter. Bayesian optimization with robust bayesian neural networks. In *Advances in Neural Information Processing Systems*, pages 4134–4142, 2016.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

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.

Jakub Tomczak and Max Welling. VAE with a VampPrior. In *International Conference on Artificial Intelligence and Statistics*, pages 1214–1223, 2018.

George Tucker, Andriy Mnih, Chris J Maddison, John Lawson, and Jascha Sohl-Dickstein. Rebar: Low-variance, unbiased gradient estimates for discrete

latent variable models. In *Advances in Neural Information Processing Systems*, pages 2627–2636, 2017.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 153–162. ACM, 2016.

Mingzhang Yin and Mingyuan Zhou. Semi-implicit variational inference. In *International Conference on Machine Learning*, pages 5646–5655, 2018.

Mingzhang Yin and Mingyuan Zhou. ARM: Augment-REINFORCE-merge gradient for stochastic binary networks. In *International Conference on Learning Representations*, 2019.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. The information autoencoding family: A Lagrangian perspective on latent variable generative models. *arXiv preprint arXiv:1806.06514*, 2018.

Jingwei Zhuo, Jun Zhu, and Bo Zhang. Adaptive dropout rates for learning with corrupted features. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.