Implicit Variational Inference with Kernel Density Ratio Fitting

Jiaxin Shi†∗, Shengyang Sun‡∗, Jun Zhu†
† Department of Computer Science & Technology, TNList Lab, Tsinghua University, Beijing, 100084
‡ Department of Electronic Engineering, Tsinghua University, Beijing, 100084
{shijx15, ssy13}@mails.tsinghua.edu.cn; dcszj@tsinghua.edu.cn

Abstract

Recent progress in variational inference has paid much attention to the flexibility of variational posteriors. Work has been done to use implicit distributions, i.e., distributions without tractable likelihoods as the variational posterior. However, existing methods on implicit posteriors still face challenges of noisy estimation and can hardly scale to high-dimensional latent variable models. In this paper, we present an implicit variational inference approach with kernel density ratio fitting that addresses these challenges. As far as we know, for the first time implicit variational inference is successfully applied to Bayesian neural networks, which shows promising results on both regression and classification tasks.

1 Introduction

Bayesian methods have been playing vital roles in machine learning by providing a principled approach for uncertainty modeling, posterior inference and preventing over-fitting [6]. As it becomes a common practice to build large and deep models that have many parameters [15], it is even more important to have a sophisticated Bayesian formulation to protect these models. For example, Bayesian Neural Networks (BayesianNN) [23, 1] have shown promise in dealing with model uncertainty and learning with few labeled data.

Besides a few simple examples, Bayesian inference is typically challenging, for which variational inference has been a standard workhorse to approximate the true posterior. Traditional variational inference focuses on mean-field variational posteriors to get analytical variational updates, i.e., the variational posterior used is from the distribution family that can be factorized across dimensions. While recent progress in this field drives variational inference into stochastic, differentiable and amortized [9, 24, 21, 13], which does not rely on analytical updates anymore, mean-field posteriors are still commonly used as the variational family. This greatly restricts the flexibility of the variational posterior, especially in high-dimensional spaces, which often leads to biased inference. There has been some works that try to improve the flexibility of variational posteriors, borrowing ideas from invertible transformation of probability distributions [26, 12, 28]. In their works, it is important for the transformation to be invertible to ensure that the transformed distribution has a tractable density and can be used as the variational posterior.

Although utilizing invertible transformation is a promising direction to increase the expressiveness of the variational posterior, we argue that a more flexible variational family can be constructed by using general deterministic or stochastic transformations, which is not necessarily invertible. As a common result, the variational posterior we get in this way does not admit a tractable likelihood, despite there is a way to sample from it. This kind of distribution is called implicit distributions, and for variational methods that use an implicit variational posterior (also known as variational programs [25], wild variational approximations [16]), we refer to them as Implicit Variational Inference (Implicit VI).

∗Equal contribution.
In this paper we present an implicit variational inference approach that uses kernel density ratio fitting, which can scale to high-dimensional latent variable models. As far as we know, for the first time implicit variational inference is successfully applied to Bayesian neural networks, which shows promising results on both regression and classification tasks.

2 Implicit variational inference

Consider a generative model with the joint distribution \( p(z, x) = p(z)p(x|z) \), where \( x \) represents observed variables, and \( z \) represents latent variables. In variational inference, a variational distribution \( q_\phi(z) \) in some parametric family is chosen to approximate the true posterior \( p(z|x) \). The objective to optimize is called the evidence lower bound (ELBO):

\[
\mathcal{L}(\phi) = \mathbb{E}_{q_\phi(z)}[\log p(x|z)] - \text{KL}(q_\phi(z) \parallel p(z)).
\]  

(1)

It can be clearly seen that this objective is a lower bound of the marginal log-likelihood \( \log p(x) \) since it can be written as

\[
\mathcal{L}(\phi) = \log p(x) - \text{KL}(q_\phi(z) \parallel p(z|x)),
\]

(2)

where \( \text{KL} \) denotes the Kullback-Leibler divergence \( \text{KL}(q \parallel p) = \mathbb{E}_q \log \frac{q}{p} \). Without extra constraint on the variational distribution, the maximum of this objective is achieved when \( q_\phi(z) = p(z|x) \). From Eq. (1), we can see that the challenge of using an implicit \( q_\phi \) is in the second term. Computing \( \text{KL}(q_\phi(z) \parallel p(z)) \) requires to evaluate the density value of \( q_\phi \), which is intractable when \( q_\phi \) is an implicit distribution.

Recently, inspired by the probabilistic interpretation of Generative Adversarial Networks [7, 22], there has been some works that extend the adversarial learning approach to the posterior inference of latent variable models [20, 10, 29]. These methods all use an implicit variational family and thus can be categorized into Implicit VI methods. One of the key observations in them is that the density ratio \( \frac{q_\phi(z)}{p(z)} \) can be estimated from samples of the two distributions by using a probabilistic classifier. They first assign class labels \( (y) \) to \( q \) and \( p \): Let samples from \( q_\phi(z) \) be of class \( y = 1 \), and samples from \( p(z) \) be of class \( y = 0 \). Given equal class priors \( p(y = 1) = p(y = 0) \), the density ratio at a given point can be calculated as

\[
\frac{q_\phi(z)}{p(z)} = \frac{p(z|y = 1)}{p(z|y = 0)} = \frac{p(y = 1|z)}{p(y = 0|z)},
\]

(3)

which is the ratio between the class probabilities given the data point. To estimate this, a probabilistic classifier \( D \) is trained to classify between the two classes, with a logistic loss:

\[
\max_D \mathbb{E}_{q_\phi(z)} \log (D(z)) + \mathbb{E}_{p(z)} \log (1 - D(z)),
\]

(4)

where \( D(z) \) is a classifier that outputs the probability of \( z \)'s being from class \( y = 1 \). Given \( D \) is flexible enough, the optimal solution of problem (4) is \( D(z) = q_\phi(z)/(q_\phi(z) + p(z)) \). Therefore, the KL divergence term in the ELBO of Eq. (1) can be approximated as

\[
\text{KL}(q_\phi \parallel p) \approx \mathbb{E}_{q_\phi(z)} \log \frac{D(z)}{1 - D(z)}.
\]

(5)

This is called prior-contrastive forms of VI in [10], and an amortized version for training local latent variable models is presented in [20]. Note that the ratio approximation doesn’t change the gradients once the approximation is accurate [10]. Gradients of the approximated ELBO are computed using the reparameterization trick [15]. Though incorporating the discriminative power in a probabilistic model has shown great success in GANs, this approach still suffers from several challenging problems when applied to variational inference:

- **Noisy density ratio estimation** In the optimization procedure of variational inference, the variational posterior gets updated in each iteration. To produce accurate estimation, the classifier should be trained to optimum after each iteration. However, due to time constraints, in practice the classifier is only trained for one or several iterations for each variational update. It is hard for the classifier to catch up with the variational posterior. This unstable loss often produces noisy gradients and leads to unsatisfying results. Besides, even if the classifier quickly achieves the
optimum in a small number of iterations, there is still another issue that contributes to the noisy estimation. Notice that the training loss in problem (4) is with expectations. But in practice we are using samples from the two distributions to approximate it. When the support of the distributions is high-dimensional, a massive number of samples are needed to ensure accurate estimates. However, given the limited number of samples we use in real problems, the variance of this estimate is considerable. In other words, the classifier may overfit the samples. The phenomenon is that the classifier achieves a state where samples are easily distinguished and the probabilities given by the classifier are near 0 or 1, which is commonly observed in experiments [20].

- High dimensional latent variables As the density ratio is estimated by a classifier, the samples of latent variables from the two distributions should be fed into it. However, the typically used neural network classifier cannot scale towards very high-dimensional inputs. People may have to resort to simple linear classifiers in order to use adversarial approaches for variational inference in this situation (e.g., moderate-size Bayesian neural networks).

To address these two challenges for Implicit VI, we propose to replace the probabilistic classifier with a kernel-based method for density ratio estimation. Specifically, when this approach is applied to Bayesian neural networks, we design a highly correlated implicit variational family that can scale to very large networks. These will be covered in the following two sections.

3 Kernel density ratio fitting

As previously mentioned, the prior-contrastive methods can be seen as an algorithm with nested loops. The outer loop is for parameter updates of the variational posterior, and the inner loop is for estimating the KL divergence term through training a probabilistic classifier to the optimum. The key problem of the noisy estimation is partly due to crude truncation of the inner loop after each variational update. Given this observation, if a method for density ratio estimation can give closed-form solutions while maintaining reasonable accuracy, it should enable an algorithm without inner loops and have better performance. Based on the intuition, we apply a kernel density ratio fitting (KDRF) method similar to the unconstrained Least Square Importance Fitting (uLSIF) [11]. Specifically, let the true density ratio be

\[ r(z) = \frac{q(z)}{p(z)}. \]  
(6)

Consider modeling the density ratio by the following linear model:

\[ \hat{r}(z) = \alpha^T \psi(z) = \sum_{k=1}^{K} \alpha_k \psi_k(z), \]  
(7)

where \( \psi_k, k \in \{1, 2, \ldots, K\} \) are the basis functions and \( \alpha_k \) are the combination weights. A common choice is to set the basis functions as RBF kernels around the training data points:

\[ \psi_k(z) = k(z_k, z) = \exp\left\{ -\frac{\|z - z_k\|^2}{2\sigma^2} \right\}, \]  
(8)

where \( \sigma \) is the kernel bandwidth. In practice, to keep the algorithm efficient, we randomly select a subset of data points from the samples of \( q \) as kernel bases. To estimate the true ratio, the following squared distance is minimized

\[ L(\alpha) = \frac{1}{2} \int (\hat{r}(z) - r(z))^2 p(z) \, dz \]  
(9)

\[ = \frac{1}{2} \mathbb{E}_p \, \hat{r}(z)^2 - \mathbb{E}_q \hat{r}(z) + C \]  
(10)

\[ = \frac{1}{2} \alpha^T H \alpha - h^T \alpha + C \]  
(11)

where \( H = \mathbb{E}_p \left[ \psi(z) \psi(z)^T \right] \), and \( h = \mathbb{E}_q \left[ \psi(z) \right] \). The expectation can be estimated directly by Monte Carlo integration using samples from the prior \( p \) and the variational posterior \( q \):

\[ \hat{H} = \frac{1}{n_p} \sum_{i=1}^{n_p} [\psi(z_i)\psi(z_i)^T], \quad z_i \sim p(z); \quad \text{and} \quad \hat{h} = \frac{1}{n_q} \sum_{j=1}^{n_q} \psi(z_j), \quad z_j \sim q_\phi(z). \]  
(12)
Adding a quadratic regularization term $\frac{\lambda}{2} \alpha^T \alpha$ to the above loss and removing the constant, we get the final objective
\[
\min_\alpha \frac{1}{2} \alpha^T \hat{H} \alpha + \hat{h}^T \alpha + \frac{\lambda}{2} \alpha^T \alpha.
\] (13)

We can get the closed-form solution as:
\[
\alpha^* = (\hat{H} + \lambda I)^{-1} \hat{h}.
\] (14)

Note that there should be a constraint that estimated density ratio should be non-negative. However, we do not involve it in the optimization objective in order to get a closed-form solution.

There are two key differences between the method we use and the original uLSIF. The first one is on how we deal with the non-negative constraint of density ratios. Because this constraint is not explicitly included in the loss objective, some post-processing must be done to ensure the estimated ratios are non-negative. In uLSIF, the authors propose to clip all $\alpha$s to be not less than zero. However, we argue that reserving some negative coefficients are essential to obtain smooth approximations, and the clipping on coefficients often leads to dramatical increase in some high density regions, resulting in over-estimates of the density ratio. We solve this issue by replacing the clipping on $\alpha$ with clipping on the estimated density ratio. The clipping values are searched from $[1e^{-8}, 1e^{-16}, 1e^{-32}]$.

The second difference is on the way of determining the kernel bandwidth. The original paper uses cross-validation, which does not fit into our settings. So we apply a heuristic approach which turns out to work very well in practice. The kernel bandwidth is chosen to be the median of all distances between the training data points and the chosen basis.

Another trick is essential to get an accurate estimate of the KL divergence. We observe that when applying the above density ratio fitting methods to estimate $\frac{q}{p}$, the optimization objective in Eq. 9 puts more weights into places where the probability mass of $p$ is high. However, the KL divergence term $KL(q\|p)$ we are estimating puts more weights into places where the probability mass of $q$ is high. Unless $p$ and $q$ match very well in where they put most probabilities, the ratio estimation objective does not fit well with the KL-divergence targets. The solution is by a simple trick. Instead of estimating $\frac{q}{p}$, we choose to estimate $\frac{p}{q}$ and compute the KL divergence term as $-E_q \log \frac{p}{q}$. We find in experiments that this trick is very essential to make the estimation accurate enough for variational inference.

Here we explain how this method addresses the two challenges stated in Section 2. First, the ratio estimates are given in closed-forms, thus not having the problem of not catching up. Second, the bias/variance trade-off of the estimation can be controlled by the regularization coefficient ($\lambda$). When $\lambda$ is set smaller, the estimation is more aggressive to match the ratios at given samples. When $\lambda$ is set larger, the estimated ratio function is smoother. Choosing the appropriate $\lambda$, the variance of estimation can be controlled while maintaining a reasonably good fit, compared to the extreme ratio estimates given by classifiers when their probabilities are near 0 or 1.

4 Implicit variational Bayesian neural networks

In a Bayesian neural network, a prior is given to the neural network parameters $W = \{W_l\}_{l=1}^L$. Given input $x$, the output $y$ is modeled with
\[
W_l \sim \mathcal{N}(W_l|0, I) ; \quad l = 1, \cdots, L
\] (15)
\[
\hat{y} = f_{NN}(x, W)
\] (16)
\[
y \sim \mathcal{P}(\hat{y}; \theta)
\] (17)

That is, $\hat{y}$ is the output of the feed forward network. And the final output $y$ is of a distribution $\mathcal{P}$ parameterized by $\hat{y}$ and $\theta$. For regression, $\mathcal{P}$ is usually a Gaussian with $\hat{y}$ as the mean. For classification, $\mathcal{P}$ is usually a discrete distribution with $\hat{y}$ as the unnormalized log probabilities.

As the true posterior of $W$ is intractable, a variational posterior $q$ can be used to approximate it by maximizing the ELBO. The variational posterior is usually set to be factorized by layer:
\[
q \left(\{W_l\}_{l=1}^L\right) = \prod_{i=1}^L q_i(W_l).
\] (18)
However, all the existing methods use a variational posterior with limited capacity, including mean-field Gaussian [8], matrix variate Gaussian [18,27], normalizing flows [19]. Enabled to learn implicit variational posterior, we propose to adopt a general distribution without explicit likelihood.

In a neural network, each layer’s parameter $W$ is very high dimensional. Therefore, we often cannot directly use a fully connected neural network to model the distribution of $W$. Drawing inspiration from low-rank matrix factorization [14], we propose a new kind of network called "Matrix Multiplication Neural Network" to model implicit distribution of a 2-D matrix, as shown in Alg. 1. In each layer of a MMNN, for input matrix $(M_{in} \times N_{in})$, we left multiply a parameter matrix $(M_{out} \times M_{in})$ and add a bias matrix $(M_{out} \times N_{in})$, then we right multiply a parameter matrix $(N_{in} \times N_{out})$ and add a bias matrix $(M_{out} \times N_{out})$. Finally it is passed through a nonlinear activation function such as Relu. We call such a layer as Matrix Multiplication Layer of size $[M_{out}, N_{out}]$.

When modeling a matrix, MMNN has significant computational advantages over fully connected networks. Due to its low-rank property, MMNN easily scales with matrix size. To model a $M \times N$ matrix, consider a one-layer network whose input shape is $[M_0, N_0]$. For fully connected structure, the parameter matrix’s shape is $[M_0 N_0, M N]$, which is extremely big. While for MMNN structure, we only need two matrices of shape $[M, M_0]$ and two matrices of shape $[N_0, N]$.

To model the implicit distribution of $W_l$ in each layer of a Bayesian neural network, we only need to randomly sample a matrix $W^0_l$ of small size $[n_{\text{particles}}, M_0, N_0]$, and feed it forward to the Matrix Multiply Neural Network to get the output matrix samples.

$$W^0_l \sim N(0, I), \quad q(W_l) = \mathcal{MMN}(\phi_l)(W^0_l) \quad (19)$$

Thus, we can use a MMNN to model the variational posterior of each layer’s parameter $W_l$ for large scale networks. However, in tasks with small network, we still use the fully connected network to model the variational posterior.

5 Related work

There are three lines of works closely related.

**Implicit generative models** Implicit generative models (generative models that define implicit distributions) have drawn much attention these days due to the popularity of Generative Adversarial Networks (GAN) [7]. General learning algorithms of implicit models have been surveyed in [22], of which density ratio estimation plays the central role. The connection between density ratio estimation and GANs is also discussed in [30].

**Variational inference** Our work builds upon the recent developments of variational inference, including stochastic optimization by mini-batches [9], direct gradient optimization of variational lower bounds [24,21], and the reparametrization trick for continuous latent variable models [13]. Following the success of learning with implicit generative models, implicit distributions are applied to variational inference, which are surveyed in [10]. This paper divides implicit variational inference into two categories: prior-contrastive and joint-contrastive. Classifiers in prior-contrastive methods distinguish between samples from the prior and the variational posterior, while in joint-contrastive methods it distinguishes between the model joint distribution and the joint distribution composed of...
data distribution and variational posteriors. Concurrent with \cite{10}, authors of \cite{20} propose Adversarial Variational Bayes, which is an amortized version of prior-contrastive methods for training local latent-variable models like VAEs \cite{13}. Prior to \cite{10}, similar ideas with joint-contrastive methods have been proposed in ALI and Bi-GAN \cite{4, 3}. Nonparametric methods for variational inference \cite{17, 2} that adapt a set of particles towards the true posterior are also closely related to implicit variational inference. They share the similar advantage of flexible approximations. As previously mentioned, there is also another line of works on flows \cite{26, 12, 28} that tries to improve the expressiveness of the variational posterior with invertible transformations.

**Bayesian neural networks** There are many recent advances in inference algorithms for Bayesian neural networks. Probabilistic back-propagation \cite{8}, structured uncertainty \cite{27} is based on Assumed Density Filtering. Bayes by backprop \cite{1}, dropout uncertainty \cite{5}, and variational matrix Gaussian \cite{18} build upon variational inference.

6 Experiments

We now present experimental results on both synthetic and real datasets to demonstrate the benefits of our method. All implementations are based on ZhuSuan \url{https://github.com/thu-ml/zhusuan}.

6.1 Synthetic data

6.1.1 Toy 1-D Gaussian mixtures

We firstly conduct a toy experiment to approximate a 1-D Gaussian mixture distribution with variational inference. The Gaussian mixture distribution has two equally distributed unit-variance components whose means are -3 and 3. We compare the results of Implicit VI (KDRF) with mean-field VI in Fig. 1. For Implicit VI (KDRF), we forward random samples from a standard normal distribution through a two-layer fully connected neural network with 50 hidden units and one output unit.

As shown in Fig. 1a, the mean-field posterior converges to the middle of the two modes, where probability mass is small. In contrast, Implicit VI (KDRF) can accurately approximate the Gaussian mixture distribution with expressive variational posterior.

6.1.2 2-D Bayesian logistic regression

We also conduct experiments on a 2-D Bayesian logistic regression example, which has an intractable posterior. The model is

\[
\begin{aligned}
\mathbf{w} & \sim \mathcal{N}(0, \mathbf{I}), \\
y_i & \sim \text{Bernoulli}(\sigma(\mathbf{w}^T \mathbf{x}_i)), \quad i = 1, \ldots, N
\end{aligned}
\]

(20)

where \( \mathbf{w}, \mathbf{x}_i \in \mathbb{R}^2; \sigma \) is the sigmoid function. We generate \( N = 200 \) data points \( \{(x_i, y_i)\}_{i=1}^{200} \) from the true model as the training data (Fig. 2a). The unnormalized true posterior is plotted in Fig. 2b. As a baseline, we first run mean-field variational inference to do posterior inference. The result is shown...
We then apply our Implicit VI method. The implicit posterior we use is a simple stochastic neural network. To see how good the result is, we also run Hamiltonian Monte Carlo (HMC) to get posterior samples. The results are plotted in Fig. 3. We can see that the implicit posterior are learned to capture the strong correlation between the two dimensions and can produce posterior samples that have a similar shape with samples drawn by HMC.

### 6.2 Bayesian neural networks

#### 6.2.1 Multivariate regression

To illustrate the predicative ability, we compare the multivariate regression results for several public datasets with Stein variational gradient descent (SVGD) [17] and dropout uncertainty [5]. Following the setup in probabilistic backpropagation [8], we randomly select 90% of the whole dataset for training and use the rest for testing and use a Bayesian neural network of one hidden layer. For the two big datasets, i.e., Protein Structure and Year Prediction MSD, the hidden layer is with 100 units. For the rest datasets, the hidden layer is with 50 units. We also put a Gamma prior on the output precision. We run 20 times and report the mean and std errors of test performances, except 5 times for Protein Structure and only 1 time for Year Prediction MSD. For information about our variational posterior network, see A.1.

The results are shown in Table 2. For each dataset, the best result is shown as bold. We can see that Implicit VI (KDRF) consistently outperforms SVGD and dropout in both RMSE and test log likelihood for most datasets. Especially in RMSE, Implicit VI (KDRF) has obvious improvements.
compared to SVGD and dropout except on Wine and Year Prediction MSD. That is mainly attributed to the feature that Implicit VI (KDRF) is enabled to learn the implicit variational posterior of the network parameters, which captures more about the complex parameter correlations. Note that VMG [18] and structured uncertainty [27] uses matrix variate Gaussian prior, which is different from our factorized prior; the former also additionally used variational dropout, thus their results are not comparable to ours.

Recently, normalizing flows are shown to have good performance on Bayesian neural networks [19]. Being interested in this, we also experiment with directly applying normalizing flows on this task. The results are reported in [4]. We apply 10 flows on the weights to match the computation time of ours. However, the results of normalizing flows do not improve over mean-field VI.

6.2.2 MNIST Classification

![Figure 4: Training lower bound in MNIST classification with prior-contrastive and Implicit VI (KDRF).](image)

Table 3: MNIST Classification error rate

| layer size | Bayes by Backprop, Gaussian | Implicit VI (KDRF) |
|------------|----------------------------|-------------------|
| 400        | 1.82%                      | 1.62%             |
| 800        | 1.99%                      | 1.53%             |
| 1200       | 2.04%                      | 1.69%             |

In this part we compare the classification results on MNIST datasets, which consists of 60000 training digit pictures and 10000 test digit pictures. As a standard classification task, MNIST classification has been worked on by many methods. However, many of them use a different prior, such as structured prior (VMG [18]), scale mixture prior (Bayes by Backprop [1]). Therefore we only compare the classification results with Bayes by Backprop (Gaussian) [1].

We use feed forward neural network with two Relu hidden layers and experiment on layer size 400, 800, 1200. For modeling implicit variational posterior, we use MMNN with two hidden Matrix Multiplication layers. For more detailed information, see [A.2]. We use randomly set learning rate 0.001, 10 random samples for testing and batch size 100 (For layer size 1200, because of memory problem we use batch size 50) and train with all 60000 data. We chose the stabling results for layer size 400 and 800. For layer size 1200 which oscillates a little, we choose the result in 200 epochs. The results are shown in [3]. As seen, with the more expressive capacity of our implicit variational posterior, Implicit VI (KDRF) show superior classification results compared to the mean field variational posterior.

To compare with prior-contrastive methods, we draw the training lower bound with time in Fig. 4. As the posterior is extremely high-dimensional, only simple logistic regression is possible for being a discriminator. So we use logistic regression as the discriminator for prior-contrastive methods. As seen, lower bounds of the two methods increase in the same pace at first, then prior-contrastive fails to converge with lower bound explosion while Implicit VI (KDRF) improves consistently. The explosion is mainly because the input to the discriminator is of hundreds of thousands of dimensions for this Bayesian neural network. Plain discriminator cannot handle with such high-dimensional inputs. We also experiment with prior-contrastive for layer size 800 and 1200. They both fail to converge at end.

7 Conclusions

We present an implicit variational inference approach with kernel density ratio fitting. This approach addresses the existing challenges of implicit variational inference, including noisy estimation and scalability with high-dimensional latent variable models. We successfully apply this approach to Bayesian neural networks and achieve superior performance on both regression and classification tasks. Future work may include applying this method on neural networks with larger scale and developing amortized version of it.
Acknowledgments

The work was supported by the National Basic Research Program (973 Program) of China (No. 2013CB329403), NSFC Projects (Nos. 61620106010, 61621136008), and the Youth Top-notch Talent Support Program.

References

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

[2] Bo Dai, Niao He, Hanjun Dai, and Le Song. Provable bayesian inference via particle mirror descent. arXiv preprint arXiv:1506.03101, 2015.

[3] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. arXiv preprint arXiv:1605.09782, 2016.

[4] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. Adversarially learned inference. arXiv preprint arXiv:1606.00704, 2016.

[5] 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, 2016.

[6] Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. Nature, 521(7553):452–459, 2015.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.

[8] Jose Miguel Hernandez-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In Proceedings of The 32nd International Conference on Machine Learning, pages 1861–1869, 2015.

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

[10] Ferenc Huszár. Variational inference using implicit distributions. arXiv preprint arXiv:1702.08235, 2017.

[11] Takafumi Kanamori, Shohei Hido, and Masashi Sugiyama. A least-squares approach to direct importance estimation. Journal of Machine Learning Research, 10(Jul):1391–1445, 2009.

[12] Diederik 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.

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

[14] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. Computer, 42(8), 2009.

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. Nature, 521(7553):436–444, 2015.

[16] Qiang Liu and Yihao Feng. Two methods for wild variational inference. arXiv preprint arXiv:1612.00081, 2016.

[17] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In Advances In Neural Information Processing Systems, pages 2370–2378, 2016.

[18] Christos Louizos and Max Welling. Structured and efficient variational deep learning with matrix gaussian posteriors. arXiv preprint arXiv:1603.04733, 2016.

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

[20] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. arXiv preprint arXiv:1701.04722, 2017.
[21] Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In Proceedings of the 31st International Conference on Machine Learning, 2014.

[22] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. arXiv preprint arXiv:1610.03483, 2016.

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

[24] John Paisley, David Blei, and Michael Jordan. Variational bayesian inference with stochastic search. arXiv preprint arXiv:1206.6430, 2012.

[25] Rajesh Ranganath, Dustin Tran, Jaan Altosaar, and David Blei. Operator variational inference. In Advances in Neural Information Processing Systems, pages 496–504, 2016.

[26] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. arXiv preprint arXiv:1505.05770, 2015.

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

[28] Jakub M Tomczak and Max Welling. Improving variational auto-encoders using householder flow. arXiv preprint arXiv:1611.09630, 2016.

[29] Dustin Tran, Rajesh Ranganath, and David M Blei. Deep and hierarchical implicit models. arXiv preprint arXiv:1702.08896, 2017.

[30] Masatoshi Uehara, Issei Sato, Masahiro Suzuki, Kotaro Nakayama, and Yutaka Matsuo. Generative adversarial nets from a density ratio estimation perspective. arXiv preprint arXiv:1610.02920, 2016.
A Networks for Modeling Variational Posteriors

A.1 Multivariate Regression

As the regression datasets have small feature dimensions (usually less than 15, except 90 for year), using Bayesian neural networks of one hidden layer (50 units) doesn’t produce very high-dimensional weights. Therefore we use two hidden layers of size 1024 and 50 (n_in + 1) with Relu activations. The input to it is random samples of length 100 from a standard normal distribution. For the second layer parameter ([1, 51]), we use two hidden layers of size 100, 51. The input to it is random samples of length 50 from a standard normal distribution.

A.2 MNIST classification

MNIST classification needs a larger scale network than the one used in multivariate regression. Therefore we use a Matrix Multiplication Neural Network for modeling the variational posterior. We experimented with hidden layer size 400, 800 and 1200, denoted by L. Below we set L = 500 when L = 400, otherwise N = 800. In the variational posterior network, we use two matrix multiplication hidden layers.

For the first layer parameter ([785, L]), the two hidden matrix multiplication layers are of size [N, N] and [N, N] with Relu activations. The final output layer is of size [L, 785] with linear activations. The input matrix is random samples of size [30, 30] from a standard normal distribution. To be noted, each MM layer represents a left multiply, a right multiply and two sums.

For the second layer parameter ([L, L + 1]), the two hidden matrix multiplication layers are of size [N, N] and [N, N] with Relu activations. The final output layer is of size [L, L + 1] with linear activations. The input matrix is random samples of size [30, 30] from a standard normal distribution.

For the third layer parameter ([10, L + 1]), the two hidden matrix multiplication layers are of size [30, N] and [30, N] with Relu activations. The final output layer is of size [10, L + 1] with linear activations. The input matrix is random samples of size [30, 30] from a standard normal distribution.

B Lower bound with Gamma-Prior Precision

In the multivariate regression task, the output is sampled from a normal distribution with \( \hat{y}(x, W) \) as mean and a parameter as variance. The variance controls the likelihood of the model, therefore, choosing an appropriate variance is essential. Therefore we place a Gamma prior \( \text{Ga}(6, 6) \) on its reciprocal (i.e., the precision of the Normal distribution). The variational posterior we used is \( q(W, \lambda) = q(W) q(\lambda) \). Then the ELBO can be computed as

\[
\mathcal{L} = E_{q(W)} E_{q(\lambda)} \log p(y | x, W, \lambda) - \text{KL} (q(W) \| p(W)) - \text{KL} (q(\lambda) \| p(\lambda))
\]

\[
= E_{q(W)} E_{q(\lambda)} \log \mathcal{N} \left( y | \hat{y}(x, W), \frac{1}{\lambda} \right) - \text{KL} (q(W) \| p(W)) - \text{KL} (q(\lambda) \| p(\lambda))
\]

\[
= \frac{1}{2} E_{q(W)} E_{q(\lambda)} \left[ \log \lambda - \lambda \left( y - \hat{y}(x, W) \right)^2 - \log 2\pi \right] - \text{KL} (q(W) \| p(W)) - \text{KL} (q(\lambda) \| p(\lambda))
\]

\[
= \frac{1}{2} E_{q(W)} \left[ \psi(\alpha) - \log \beta - \frac{\alpha}{\beta} (y - \hat{y}(x, W))^2 - \log 2\pi \right] - \text{KL} (q(W) \| p(W)) - \text{KL} (q(\lambda) \| p(\lambda)).
\]

Where \( \psi(x) \) is the digamma function and the KL divergence of \( \lambda \) can be calculated in closed-form.