Federated Functional Variational Inference

Michael Hutchinson *
Department of Statistics
University of Oxford
hutchinson.michael.john@gmail.com

Matthias Reisser
Qualcomm AI Research
Qualcomm Technologies Netherlands B.V.
mreisser@qti.qualcomm.com

Christos Louizos
Qualcomm AI Research
Qualcomm Technologies Netherlands B.V.
clouizos@qti.qualcomm.com

Abstract

Traditional federated learning involves optimizing point estimates for the parameters of the server model, via a maximum likelihood objective. Models trained with such objectives show competitive predictive accuracy, however they are poorly calibrated and provide no reliable uncertainty estimates. These, however, are particularly important in safety critical applications of federated learning, such as self-driving cars and healthcare. In this work, we propose `FSVI`, a method to train Bayesian neural networks in the federated setting. Bayesian neural networks provide a distribution over the model parameters, which allows to obtain uncertainty estimates. Instead of employing prior distributions and doing inference over the model parameters, `FSVI` builds upon recent advances in functional variational inference and posits prior distributions directly in the function space of the network. We discuss two different approaches to federated `FSVI`, based on `FedAvg` and model distillation respectively, and show its benefits compared to traditional weight-space inference methods.

1 Method

In this paper we are concerned with neural networks, functions of data and parameters to outputs, $f : \mathcal{X} \times \Theta \to \mathcal{Y}$, and supervised learning tasks where from some set of observations $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, $\mathbf{x}_i \in \mathcal{X},\ \mathbf{y}_i \in \mathcal{Y}$ we want to learn the optimal set of parameters. We denote the function $f(\cdot, \boldsymbol{\theta}) : \mathcal{X} \to \mathcal{Y}$ as $f; \boldsymbol{\theta}$. In the probabilistic setting we posit a prior distribution over the parameters $\boldsymbol{\theta}$, governed by a set of hyperparameters $\boldsymbol{\nu}$, $p_{\boldsymbol{\nu}}(\boldsymbol{\theta})$; this induces a specific prior distribution over functions $p_{\boldsymbol{\nu}}(f; \boldsymbol{\theta})$. The notation $p_{\boldsymbol{\nu}}(\theta)$ describes a distribution over $\boldsymbol{\theta}$, parametrised by hyperparameters $\boldsymbol{\nu}$. Where it is clear, we will suppress the hyperparameter notation. Given an observed dataset $\mathcal{D}$ and a likelihood function $l_{\mathcal{D}}(f)$ (typically $l_{\mathcal{D}}(f) = \prod_{\{\boldsymbol{x},\boldsymbol{y}\}\in\mathcal{D}} p(\boldsymbol{y}|\boldsymbol{x}, f)$, which maps a function to the likelihood of the data under said function, we are then interesting in obtaining the *posterior* distribution over the parameters, given the observed data

$$p_{\boldsymbol{\nu}}(\boldsymbol{\theta}|\mathcal{D}) = \frac{l_{\mathcal{D}}(f; \boldsymbol{\theta}) p_{\boldsymbol{\nu}}(\boldsymbol{\theta})}{\int l_{\mathcal{D}}(f; \boldsymbol{\theta}) p_{\boldsymbol{\nu}}(\boldsymbol{\theta}) d\boldsymbol{\theta}} \tag{1}$$

*Work done while interning at Qualcomm AI Research. Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

Bayesian Deep Learning workshop, NeurIPS 2021.

We can then use this posterior distribution in order to obtain a distribution over predictions $\mathbf{y}^*$ conditioned on a new input $\mathbf{x}^*$ and the observed data $\mathcal{D}$

$$p_{\boldsymbol{\nu}}(\mathbf{y}^*|\mathbf{x}^*, \mathcal{D}) = \int_{\boldsymbol{\theta}\in\Theta} p(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\theta}) p_{\boldsymbol{\nu}}(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} = \int_{\boldsymbol{\theta}\in\Theta} l_{(\boldsymbol{x}^*, \boldsymbol{y}^*)}(f; \boldsymbol{\theta}) p_{\boldsymbol{\nu}}(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} \tag{2}$$

1.1 Variational Inference

Weight Space Variational Inference (`WSVI`) Typically computing the distribution $p_{\boldsymbol{\nu}}(\boldsymbol{\theta}|\mathcal{D})$ is intractable. Variational inference (Blei et al., 2017) approximates these distributions by first choosing a parametric family of *variational* distributions, $\mathcal{Q}$, and then finding the member of the variational family that minimises the Kullback–Leibler (KL) divergence between the variational distribution and the true posterior over parameters, $\mathbb{D}_{\text{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\nu}}(\boldsymbol{\theta}|\mathcal{D})\right)$. This term is typically decomposed to make the objective computationally tractable as

$$\mathbb{D}_{\text{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\nu}}(\boldsymbol{\theta}|\mathcal{D})\right) = -\mathbb{E}_{\boldsymbol{\theta}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta})}\left[l_{\mathcal{D}}(f; \boldsymbol{\theta})\right] + \mathbb{D}_{\text{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\nu}}(\boldsymbol{\theta})\right) + \underbrace{\log p_{\boldsymbol{\nu}}(\mathcal{D})}_{const.} \tag{3}$$

where $p_{\boldsymbol{\nu}}(\mathcal{D}) = \int l_{\mathcal{D}}(f; \boldsymbol{\theta}) p_{\boldsymbol{\nu}}(\boldsymbol{\theta}) d\boldsymbol{\theta}$, the log model evidence. Thus we can see that minus the first two terms of eq. (3) gives a lower bound on the log model evidence. The first term can be approximated via Monte Carlo methods and the second term is analytic if we pick a suitable prior and variational family. A large body of work performing variational inference on the parameters of neural networks exists, with mixed success, particularly on larger networks and tasks.

Function Space Variational Inference (`FSVI`) Recent work (Burt et al., 2020) has however shown that in more complex models such as neural networks, it is not the weights of the model we should be performing variational inference on, but the *functions* these weights encode. Working with spaces of functions is however non-trivial from a probabilistic perspective, and requires the tools of measure theory (see appendix B for an introduction). Beginning with the KL divergence between a parametric variational *measure* over functions $Q_{\boldsymbol{\phi}}$ and the posterior *measure* over functions $P_{\mathcal{D}}$, $\mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\phi}}||P_{\mathcal{D}}\right)$, we arrive at a familiar looking objective

$$\mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\phi}}||P_{\mathcal{D}}\right) = -\mathbb{E}_{f\sim Q_{\boldsymbol{\phi}}}\left[l_{\mathcal{D}}(f)\right] + \mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\phi}}||P_{\boldsymbol{\nu}}\right) + \underbrace{\int l_{\mathcal{D}}(f) dP_{\boldsymbol{\nu}}}_{const.} \tag{4}$$

The main difference now is that $\mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\phi}}||P_{\boldsymbol{\nu}}\right)$ is a KL divergence between measures on function spaces, which are in general not analytic. We follow Rudner et al., 2021 in their approach to approximating this term, although others exists (e.g. Sun et al., 2019, see appendix C.1 for details).

1.2 Federated Learning (FL)

In the Federated Learning (FL) setting we typically have a a series of clients, each with a shard of data $\mathcal{D}_s$. The objective is to learn from this local data centrally without accessing it directly to preserve privacy and reduce communication costs (Kairouz et al., 2021).

Federated Averaging (`FedAvg`) (McMahan et al., 2017) is the standard method for training neural networks in FL. It is applicable to problems with a global loss functions of the form

$$\min_{w\in\mathbb{R}^d} f(w) \quad \text{where} \quad f(w) = \frac{1}{N}\sum_{i=1}^{N} f_i(w). \tag{5}$$

A set of clients are selected at each round of the algorithm. Each client is sent the server model and updates the model using its local data (typically by a variant of SGD). The clients return these updated models to the server, and the server model is updated to the average of these weights.

Graphical model perspective on federated learning An alternative view of `FedAvg` is given by Louizos et al., 2021, which formulates federated learning as a hierarchical model, with a set of central server parameters $\boldsymbol{\theta}$ and latent client parameters $\boldsymbol{\theta}_s$. The server parameters are learnt by

optimising the data log-likelihood, with the client parameter distributions marginalised out.

$$\boldsymbol{\nu}^* = \arg\max_{\boldsymbol{\nu}} \sum_{s=1}^{S} \log p(\mathcal{D}_s|\boldsymbol{\nu}) = \arg\max_{\boldsymbol{\nu}} \sum_{s=1}^{S} \log \int p(\mathcal{D}_s|\boldsymbol{\theta}_s) p(\boldsymbol{\theta}_s|\boldsymbol{\nu}) d\boldsymbol{\theta}_s. \tag{6}$$

For optimization they introduce a variational distribution on each clients' weights with hyper parameters $\boldsymbol{\nu}_s$ and apply the EM algorithm.

$$\sum_{s=1}^{S} \log \int p(\mathcal{D}_s|\boldsymbol{\theta}_s) p(\boldsymbol{\theta}_s|\boldsymbol{\nu}) d\boldsymbol{\theta}_s \geq \sum_{s=1}^{S} \mathbb{E}_{\boldsymbol{\theta}_s \sim q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)} \left[\log p(\mathcal{D}_s|\boldsymbol{\theta}_s) + \log p(\boldsymbol{\theta}_s|\boldsymbol{\nu}) - \log q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)\right]. \tag{7}$$

By adopting a Gaussian prior for $p(\boldsymbol{\theta}_s|\boldsymbol{\nu})$ and optimising this latent variable model with the hard version of the Expectation-Maximisation (EM) one can obtain `FedAvg`.

1.3 Five algorithms for Federated Bayesian Learning

Here we present five new options for performing Federated Bayesian inference, three of with focus on performing inference in function space rather than weight space.

1. Federated Variational EM Weight-space VI (`FedVEM WSVI`) Louizos et al., 2021 focus on performing inference in typical ML models with point estimate weights. It is simple to extend this framework to have a server model with parameter uncertainty. We achieve this by setting $\boldsymbol{\nu} = [\boldsymbol{\mu}, \boldsymbol{\sigma}]$, $p(\boldsymbol{\theta}_s|\boldsymbol{\nu}) = \mathcal{N}(\boldsymbol{\theta}_s; \boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ and performing variational EM, *i.e.*, doing variational inference for $\boldsymbol{\theta}_s$ as done by Blundell et al., 2015, at each shard with $p(\boldsymbol{\theta}_s|\boldsymbol{\nu})$ as the prior. The local and server update rules are derived in Appendix D.2. We refer to this method as `FedVEM WSVI`.

2/3. Federated Averaging Weight-space / Function-space VI (`FedAvg WSVI`, `FedAvg FSVI`) Both, `WSVI` and `FSVI` objectives can be written in the form required to apply `FedAvg` (McMahan et al., 2017), *i.e.* a sum of local objective functions (*c.f.* Appendices D.1 and C.4). It is therefore relatively simple to apply `FedAvg` in order to perform federated Bayesian inference. These methods are referred to as `FedAvg WSVI`, `FedAvg FSVI` in the rest of the paper.

4. Federated Variational EM Function-space VI (`FedVEM FSVI`) Here we derive a new objective function for federated Bayesian learning from the functional perspective, equivalent to performing variational EM on a latent *function* hierarchical model. Similar to `FedVEM WSVI` we propose a series of local variational function distributions, $Q_{\boldsymbol{\nu}_s}$ and a single global function distribution, $P_{\boldsymbol{\nu}}$, where $\boldsymbol{\nu}, \boldsymbol{\nu}_s$ are sets of hyper-parameters defining the probability measures. The single global model forms a prior for the local models. The algorithm consists of alternating optimisation of

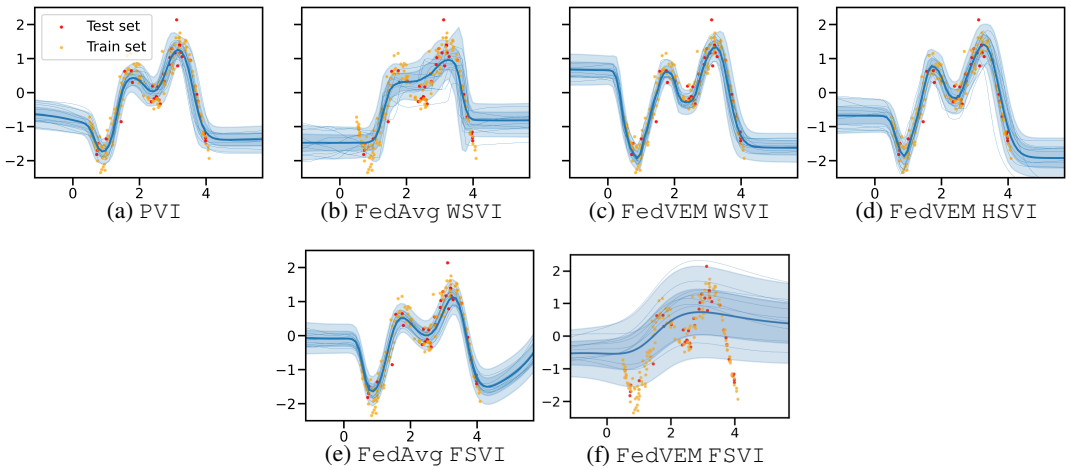
1. The local function distributions, given the global prior.
2. The global prior, to better match the local function distributions.

To derive the objective we use, we begin with the KL divergence between the joint local distributions, $Q = \prod_{s=1}^{S} Q_{\boldsymbol{\nu}_s}$ and the true posterior of the local functions given the data, $P_{\mathcal{D}}$, $\mathbb{D}_{\text{KL}}\left(Q||P_{\mathcal{D}}\right)$. Note this is the joint posterior over *all* local functions, given *all* the local data. Assuming that each local function is independent of all the data except that shard's local data, we get

$$\mathbb{D}_{\text{KL}}\left(Q||P_{\mathcal{D}}\right) = \sum_{s=1}^{S} \mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\nu}_s}||P_{\boldsymbol{\nu}}\right) - \mathbb{E}_{f\sim Q_{\boldsymbol{\nu}_s}}\left[\log l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})\right] + \underbrace{\log \int l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s}) dP_{\boldsymbol{\nu}}(f_{\boldsymbol{X}_s})}_{const.}, \tag{8}$$

where $f_{\boldsymbol{X}_s}$ is the function evaluated at the local data input locations. Rearranging gives the desired lower bound. The full derivation can be found in Appendix C.2. We optimise this objective by alternating between optimising $\boldsymbol{\nu}, \boldsymbol{\nu}_s$, and apply the results of Rudner et al., 2021 to approximate the first term. For the rest of the paper this method will be referred to as `FedVEM FSVI`.

5. `FedVEM FSVI` with global weight-space objective During the course of experimentation, we observed particular difficulty with the server KL matching term in the `FedVEM FSVI` algorithm. An alternative proposal is to perform the local inference in function space and do the server KL matching in weight space. This objective is still maximizing a lower bound on the marginal data likelihood (Appendix C.3). This algorithm is referred to as `FedVEM HSVI` (with `HSVI` standing for Hybrid-Space Variational Inference).



(a) PVI (b) FedAvg WSVI (c) FedVEM WSVI (d) FedVEM HSVI

(e) FedAvg FSVI (f) FedVEM FSVI

Figure 1: Example fits to the toy regression dataset

Table 1: Comparison of in- and out-of-distribution performance metrics (mean ± standard error over random seeds) for MNIST. AUROC is measured based on entropy.

Method	**Accuracy** ↑	**NLL** ↓	**ECE** ↓	**OOD-AUROC** ↑
PVI	$99.28_{\pm 0.01}$	$0.0224_{\pm 0}$	$0.37_{\pm 0.01}$	99.32
FedAvg WSVI	$99.38_{\pm 0.02}$	$0.0248_{\pm 0}$	$0.69_{\pm 0.03}$	99.46
FedVEM WSVI	$99.30_{\pm 0.01}$	$0.0240_{\pm 0}$	$0.16_{\pm 0.01}$	98.39
FedAvg FSVI	$99.21_{\pm 0.05}$	$0.0699_{\pm 0.0278}$	$3.98_{\pm 2.57}$	99.61
FedVEM HSVI	$99.34_{\pm 0.03}$	$0.0284_{\pm 0.0028}$	$0.88_{\pm 0.26}$	99.33

2 Experiments

We evaluate the described methods against PVI (Bui et al., 2018) on two toy federated learning tasks. We show trade-offs between weight-space and function-space VI and discuss the shortcomings of each. In-depth descriptions of the model architectures and training procedures used can be found in Appendix E.2.

2.1 Toy regression task

A 1d regression tasks allows to visually access the fit of the trained models as well as their uncertainty estimates intuitively. Figure 1 shows the training and test datasets (details in Appendix E.2), as well as samples from the individual trained models. We plot the mean plus/minus one standard deviation of 100 functions sampled from the learned posteriors, as well as 10 random functions.

Most of the methods presented here perform reasonably well from a qualitative point of view (Table 3 provides more quantitative metrics). The two exceptions to this are FedAvg WSVI, and FedVEM FSVI. The performance of FedAvg WSVI may be due to tuning issues, although similar amounts of time and practises were spent tuning the algorithms. The performance of FedVEM FSVI is drastically worse, but comes with a much clearer explanation. Figure 2 shows the global posterior distribution, with the local latent client distribution overlaid. It is clear that the server matching step is not performing its function as intended. This remained despite significant tuning, and we believe is due to optimisation difficulties of this objective, not a methodological flaw.

2.2 MNIST

Table 1 shows the performance of weight-space and function-space methods on MNIST. Accuracy, negative log-likelihood (NLL) and expected calibration error (Guo et al., 2017) are evaluated on

the test-set. All metrics are computed based on 1000 samples from the posterior, *i.e.* by averaging the softmax-outputs. OOD-AUROC is computed based on the entropy of $p(y|x)$ and we report the average across three out of distribution datasets: Fashion-MNIST (Xiao et al., 2017), standard uniform and standard normal noise. Detailed results can be found in Appendix F.2. We are interested in good predictive accuracy on the in-distribution test-set, while being well-calibrated and assigning higher entropy to out of distribution data.

We see that `FedVEM WSVI` is better calibrated than `FedVEM HSVI`, while `FedVEM HSVI` outperforms in likelihood. `FedVEM FSVI` has low likelihood and is badly calibrated, pointing to issues similar to what we observed in the regression setting. Mainly our results confirm that MNIST is not the right dataset to draw final conclusions from and serves as a proof-of-concept at most.

3 Conclusion

Our investigation into applying function-space variational inference to the federated learning setting show promising initial results. Optimizing the function-space KL in `FSVI` with respect to the prior appears to be challenging, as we encountered difficulties in `FedVEM FSVI`. We aim to further develop and evaluate the `HSVI` approach on more challenging datasets, continue investigating the challenges with `FedVEM FSVI`and explore better strategies for inducing-point selection. The latter is of particular importance in the federated setting.

References

S. Ahn, B. Shahbaba, and M. Welling. Distributed Stochastic Gradient MCMC. In *Proceedings of the 31st International Conference on Machine Learning*, 2014. Cited on page 7.

R. Bardenet, A. Doucet, and C. Holmes. On Markov chain Monte Carlo methods for tall data. *Journal of Machine Learning Research*, 18:1–43, 2017. Cited on page 7.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017. Cited on pages 2, 8.

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622, 2015. Cited on page 3.

T. D. Bui, C. V. Nguyen, S. Swaroop, and R. E. Turner. Partitioned Variational Inference: A unified framework encompassing federated and continual learning, 2018. arXiv: `1811.11206 [stat.ML]`. Cited on pages 4, 7.

D. R. Burt, S. W. Ober, A. Garriga-Alonso, and M. van der Wilk. Understanding Variational Inference in Function-Space. In *Third Symposium on Advances in Approximate Bayesian Inference*, Nov. 2020. Cited on pages 2, 8.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017. Cited on page 4.

L. Hasenclever, S. Webb, T. Lienart, S. Vollmer, B. Lakshminarayanan, C. Blundell, and Y. W. Teh. Distributed Bayesian Learning with Stochastic Natural Gradient Expectation Propagation and the Posterior Server. *Journal of Machine Learning Research*, 18:1–37, 2017. Cited on page 7.

P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konečný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao. Advances and Open Problems in Federated Learning, 2021. arXiv: `1912.04977 [cs.LG]`. Cited on page 2.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization, 2017. arXiv: `1412.6980 [cs.LG]`. Cited on page 13.

D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. *Advances in neural information processing systems*, 2015. Cited on page 11.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. Cited on page 13.

D. Levi, L. Gispan, N. Giladi, and E. Fetaya. Evaluating and Calibrating Uncertainty Prediction in Regression Tasks, 2020. arXiv: 1905.11659 [cs.LG]. Cited on page 13.

C. Louizos, M. Reisser, J. Soriaga, and M. Welling. A Graphical Model Perspective on Federated Learning, 2021. URL: https://dp-ml.github.io/2021-workshop-ICLR/files/34.pdf. Cited on pages 2, 3, 11.

A. G. d. G. Matthews, J. Hensman, R. Turner, and Z. Ghahramani. On Sparse Variational Methods and the Kullback-Leibler Divergence between Stochastic Processes. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, 2016. Cited on page 9.

B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 2017. Cited on pages 2, 3.

S. Minsker, S. Srivastava, L. Lin, and D. B. Dunson. Scalable and robust Bayesian inference via the median posterior. In *31st International Conference on Machine Learning, ICML 2014*, 2014. Cited on page 7.

W. Neiswanger, C. Wang, and E. P. Xing. Asymptotically exact, embarrassingly parallel MCMC. In *Uncertainty in Artificial Intelligence - Proceedings of the 30th Conference*, 2014. Cited on page 7.

Y. Polyanskiy and Y. Wu. Lecture notes on information theory, 2014. Cited on page 8.

S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan. Adaptive Federated Optimization, 2021. arXiv: 2003.00295 [cs.LG]. Cited on page 12.

T. G. J. Rudner, Z. Chen, and Y. Gal. Rethinking Function-Space Variational Inference in Bayesian Neural Networks. In *Third Symposium on Advances in Approximate Bayesian Inference*, 2021. Cited on pages 2, 3, 8, 10.

S. L. Scott, A. W. Blocker, F. V. Bonassi, H. A. Chipman, E. I. George, and R. E. McCulloch. Bayes and Big Data: The Consensus Monte Carlo Algorithm. *International Journal of Management Science and Engineering Management*, 11:78–88, 2016. Cited on page 7.

S. Srivastava, V. Cevher, Q. Tran-Dinh, and D. B. Dunson. WASP: Scalable Bayes via barycenters of subset posteriors. In *Journal of Machine Learning Research*, volume 38, pages 912–920, 2015. Cited on page 7.

S. Sun, G. Zhang, J. Shi, and R. Grosse. FUNCTIONAL VARIATIONAL BAYESIAN NEURAL NETWORKS. In *International Conference on Learning Representations*, 2019. Cited on pages 2, 8.

X. Wang and D. B. Dunson. Parallelizing MCMC via Weierstrass Sampler, 2014. arXiv: 1312.4605 [stat.CO]. Cited on page 7.

M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th international conference on machine learning*, 2011. Cited on page 7.

S. R. White, T. Kypraios, and S. P. Preston. Piecewise Approximate Bayesian Computation: fast inference for discretely observed Markov models using a factorised posterior distribution. *Statistics and Computing*, 25(2):289–301, 2013. Cited on page 7.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. Aug. 28, 2017. arXiv: cs.LG/1708.07747 [cs.LG]. Cited on page 5.

A Related Work

While there exist no prior work on function-space inference in the federated learning setting, there exist a number of prior weight space inference methods.

One popular line of work involves factorising the posterior over the data, and performing some type of MCMC based sampling of these sub-posteriors, before fusing them back together (Bardenet et al., 2017; Scott et al., 2016; Neiswanger et al., 2014; White et al., 2013; Wang and Dunson, 2014; Minsker et al., 2014; Srivastava et al., 2015). These methods are typically employed in statistical inference problems with tall data, but do not scale particularly well to models the size of modern neural networks.

Ahn et al., 2014 develop a distributed version of stochasitc gradient Langevin dynamics (SGLD, Welling and Teh, 2011), although do not apply the method to neural networks.

The works of Bui et al., 2018 and Hasenclever et al., 2017 are the most applicable to federated inference in BNNs. Based on Variational Inference and Expectation propagation respectively, both methods are weight-space inference methods, and rely on the existence of exponential families of distributions over the space one is performing inference over, and so do not readily extend to the functional setting.

B Measure-theoretic preliminaries

Through this paper we will also need some basic tools of measure theory, a rigorous formalisation of probability, and so we briefly introduce these. A probability measure P is a function from a specific set of subsets of some space $\mathcal{X}$ (this set of subsets is a sigma-algebra) into the range $[0, 1]$. The probability of a given given event E is given by $P(E)$ (as long as E is in the sigma-algebra). A random variable Z is said to be distributed P, $Z \sim P$ if for all events (in the sigma algebra) $\Pr[Z \in E] = P(E)$. For two measures, P, Q, defined on the same space, we say Q is *absolutely continuous* with respect to P, $Q \ll P$, if for all events (in the sigma algebra) $P(E) = 0 \implies Q(E) = 0$. Note it is possible to have $P \ll Q \ll P$. If we have $Q \ll P$, then there exists some function f such that

$$Q(E) = \int_E f dP \tag{9}$$

This function is called the Radon-Nikodym derivative, and is usually denoted $f = \frac{dQ}{dP}$. $\frac{dQ}{dP}$ is also called the density of Q with respect to P. In most of statistics, the space on which our random variables live is some subset of $\mathbb{R}^k$, for $k < \infty$. For such spaces we can choose a particular *base measure*, i.e. a measure which we will compare all other measures to, called the *Lebesgue measure*, which for $k = 1, 2, 3$ coincides with the standard notion of length, area and volume, and generalises to higher k. Typically the Lebesgue measure is denoted dx, and so the density of a measure with respect to the Lebesgue measure is the function such that $P(E) = \int_E p(x)dx$, which is recognisable as the typical definition of a probability density. Once we have these densities with respect to the same base measure, we can begin to manipulate them in the normal ways. When the densities of two measures exist with respect to some common base measure then their Radon-Nikodym derivative is simply the ratio of these densities, $\frac{dQ}{dP}(x) = \frac{q(x)}{p(x)}$.

The difficulty when working with probabilities on spaces of functions arises as, for example, the space of functions $f : \mathbb{R} \to \mathbb{R}$ is equivalent to $\mathbb{R}^{\mathbb{R}}$, i.e. $k \not< \infty$. For these spaces there is no notion of a Lebesgue measure, and so the choice of base measure to use becomes unclear. Instead we fall back to using measure theory tools.

C Functional Variational Inference

C.1 Introduction to Functional Variational Inference

While the weights of a neural network are what define the function that it represents, they are not in fact the thing we are interested in obtaining when performing inference on a neural network. We are instead interested just in the functions that these parameters encode. In many situations, such in linear regression, performing variational inference on the parameters versus on the functions

the parameters encode turns out to be the same thing (Burt et al., 2020). However, due to the significant symmetry present in the parametrisation of neural networks, and the resulting invariance to parameter transformations, in the case of neural networks variational inference on the parameters and the functions encoded is not the same.

To make this more concrete: Given a random variable $Z \sim P$, we can transform this into a new random variable by transforming it by a function g to give $g(Z)$. The distribution of this new random variable is denoted g_*P, the *pushforwad measure* of P by g.

The data processing inequality tells us that if we transform two measures in this way, they cannot become more easy to tell apart. [Data processing inequality, Polyanskiy and Wu, 2014, Thm 6.2] Let g be a measurable function. Then for two measures P, Q

$$\mathbb{D}_{\mathrm{KL}}\left(g_*Q||g_*P\right) \leq \mathbb{D}_{\mathrm{KL}}\left(Q||P\right) \tag{10}$$

with equality if and only if g is an injection. The immediate consequence of this is that we see that measuring the KL divergence between distributions of weights in a neural network is distinctly different to measuring the KL divergence between the pushforward measures on the space of functions that these weights encode. We therefore would prefer to perform variational inference on distributions in functions space, rather than weight space.

Starting from the measure theoretic definition of the log-likelihood, we can derive the measure theoretic version of variational inference (Burt et al., 2020; Blei et al., 2017). Assuming $f \sim P$, some stochastic process, defining $l_{\mathcal{D}}$ as the likelihood function of the data and $P_{\mathcal{D}}$ as the posterior distribution of f given the data D, and given a variational distribution $Q_{\boldsymbol{\phi}} \in \mathcal{Q}$,

$$\log \int l_{\mathcal{D}}(f) dP = \mathbb{D}_{\mathrm{KL}}\left(Q||P_{\mathcal{D}}\right) + \int \log l_{\mathcal{D}}(f) dQ - \mathbb{D}_{\mathrm{KL}}\left(Q||P\right) \tag{11}$$

$$\geq \int \log l_{\mathcal{D}}(f) dQ - \mathbb{D}_{\mathrm{KL}}\left(Q||P\right) \tag{12}$$

As in the weight space case, this first term can be Monte Carlo approximated as long as $l_{\mathcal{D}}(f)$ is analytic and we can sample from Q. The second term however is no longer easy to make analytic.

Two approaches have recently been proposed to approximating this term. Sun et al., 2019 prove the following [Sun et al., 2019, Theorem 1] For measures P, Q on (the product sigma-algebra) of $\mathbb{R}^{\mathcal{X}}$,

$$\mathbb{D}_{\mathrm{KL}}\left(Q||P\right) = \sup_{\boldsymbol{X} \subset \mathcal{X}, |\boldsymbol{X}| < \infty} \mathbb{D}_{\mathrm{KL}}\left(Q_{\boldsymbol{X}}||P_{\boldsymbol{X}}\right) \tag{13}$$

where Q_X, P_X are the marginals of P, Q of the measures Q and P on the set X. In the objective function Sun et al., 2019 replace this $\sup$ with an expectation over some distribution over index sets and optimise the modified objective.

In contrast Rudner et al., 2021 approximate this divergence by linearising the stochastic function around the mean of the weight distribution, and making prior conditional matching and posterior marginal consistency assumptions, allowing for the evaluation of the divergence on a finite set of index points. [Rudner et al., 2021, Proposition 1] For a mapping f of stochastic parameters $\boldsymbol{\theta}$ with mean $\boldsymbol{m}$, covariance $\boldsymbol{\Sigma}$, and Jacobian of the function at the mean parameters $\mathcal{J}_{\boldsymbol{m}}(\cdot) = \frac{\partial f(\cdot;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}|_{\boldsymbol{\theta}=\boldsymbol{m}}$, the linearisation of the stochastic function f is given by

$$f(\cdot;\boldsymbol{\theta}) \approx \tilde{f}(\cdot,\boldsymbol{\theta}) = f(\cdot,\boldsymbol{m}) + \mathcal{J}_{\boldsymbol{m}}(\cdot)(\boldsymbol{\theta} - \boldsymbol{m}) \tag{14}$$

The distribution over the linearised mapping evaluated at a vector of inputs $\boldsymbol{X} \in \mathcal{X}^n$ is given by

$$p(\tilde{f}(\boldsymbol{X};\boldsymbol{\theta})) = \mathcal{N}(f(\boldsymbol{X};\boldsymbol{m}), \mathcal{J}_{\boldsymbol{m}}(\boldsymbol{X})\boldsymbol{\Sigma}\mathcal{J}_{\boldsymbol{m}}(\boldsymbol{X})^\top) \tag{15}$$

For a variational distribution $q(\boldsymbol{\theta})$ and a prior distribution $p(\boldsymbol{\theta})$ over the weights $\boldsymbol{\theta}$, under additional assumptions and approximations (see Rudner et al., 2021 Appendix B) we obtain the following approximation to eq. (12)

$$q^*(\boldsymbol{\theta}) = \arg\max_{q(\boldsymbol{\theta}) \in Q_{\boldsymbol{\theta}}} \sum_{i=1}^{N} \mathbb{E}_{q(f;\boldsymbol{\theta})}\left[\log p(\mathbf{y}_i|\mathbf{x}_i,\boldsymbol{\theta})\right] - \mathbb{D}_{\mathrm{KL}}\left(\tilde{q}\left(\tilde{f}\left(\boldsymbol{X}_I;\boldsymbol{\theta}\right)\right)\middle|\middle|\tilde{p}\left(\tilde{f}\left(\boldsymbol{X}_I;\boldsymbol{\theta}\right)\right)\right) \tag{16}$$

where

$$\tilde{q}(\tilde{f}(\boldsymbol{X};\boldsymbol{\theta})) = \mathcal{N}(f(\boldsymbol{X};\boldsymbol{m}), \mathcal{J}_{\boldsymbol{m}}(\boldsymbol{X})\boldsymbol{\Sigma}\mathcal{J}_{\boldsymbol{m}}(\boldsymbol{X})^\top) \tag{17}$$

$$\tilde{p}(\tilde{f}(\boldsymbol{X};\boldsymbol{\theta})) = \mathcal{N}(f(\boldsymbol{X};\boldsymbol{m}_0), \mathcal{J}_{\boldsymbol{m}_0}(\boldsymbol{X})\boldsymbol{\Sigma}_0\mathcal{J}_{\boldsymbol{m}_0}(\boldsymbol{X})^\top) \tag{18}$$

$\boldsymbol{m}, \boldsymbol{\Sigma}$ are the variational weight distribution parameters, $\boldsymbol{m}_0, \boldsymbol{\Sigma}_0$ are the prior weight distribution parameters.

C.2 Functional Variational EM Objective

Consider the space of functions $\mathcal{F} = f : \mathcal{X} \to \mathbb{R}$, let there be a sigma algebra on this, $\sigma_{\mathcal{F}}$ such that we get a measurable space $(\mathcal{F}, \sigma_{\mathcal{F}})$. Next, consider the product space $\mathcal{F}^S = \prod_{s=1}^S \mathcal{F}$, the product space of measurable spaces over functions that is the measure space for the latent functions. We now define:

1. $P_{\boldsymbol{\nu}}^S$, a prior measure over local functions. This is a product of the same of the prior over each space, $P_{\boldsymbol{\nu}}^S = \prod_{s=1}^S P_{\boldsymbol{\nu}}$
2. $P_{\mathcal{D}}$, the posterior measure over local functions. We assume that each local posterior is independent of all data except the relevant local data, giving $P_{\mathcal{D}} = \prod_{s=1}^S P_{\mathcal{D}_s}$.
3. Q, the approximate measure over local functions. By again following independence, we get $Q = \prod_{s=1}^S Q_{\boldsymbol{\nu}_s}$, the product of local client function measures.

Next we minimise the following KL divergence:

$$\begin{aligned}
&\mathbb{D}_{\mathrm{KL}}\left(Q||P_{\mathcal{D}}\right) && \text{Definition of KL}\\
&= \int_{\mathbb{R}^X} \log \frac{d\prod_{s=1}^S Q_{\boldsymbol{\nu}_s}}{d\prod_{s=1}^S P_{\mathcal{D}_s}}(f) d\prod_{s=1}^S Q_{\boldsymbol{\nu}_s}(f) && \text{Plug in the product measures}\\
&= \int_{\mathbb{R}^X} \log \prod_{s=1}^M \frac{dQ_{\boldsymbol{\nu}_s}}{dP_{\mathcal{D}_s}}(f) \prod_{s=1}^M dQ_{\boldsymbol{\nu}_s}(f) && \text{Product measure decomposition and RN derivatives}\\
&= \sum_{s=1}^S \int_{\mathbb{R}^X} \log \frac{dQ_{\boldsymbol{\nu}_s}}{dP_{\mathcal{D}_s}}(f) \prod_{s=1}^S Q_{\boldsymbol{\nu}_s}(f) && \text{Log rules + linearity of integration}\\
&= \sum_{s=1}^S \int_{\mathbb{R}^X} \log \frac{dQ_{\boldsymbol{\nu}_s}}{dP_{\mathcal{D}_s}}(f) dQ_{\boldsymbol{\nu}_s}(f) && \text{Measures integrate away}\\
&= \sum_{s=1}^S \int_{\mathbb{R}^X} \log \frac{dQ_{\boldsymbol{\nu}_s}}{dP_{\boldsymbol{\nu}}}(f) \frac{dP_{\boldsymbol{\nu}}}{dP_{\mathcal{D}_s}}(f) dQ_{\boldsymbol{\nu}_s}(f) && \text{RN derivative chain rule}\\
&= \sum_{s=1}^S \int_{\mathbb{R}^X} \log \frac{dQ_{\boldsymbol{\nu}_s}}{dP_{\boldsymbol{\nu}}}(f) dQ_{\boldsymbol{\nu}_s}(f) - \int_{\mathbb{R}^X} \log \frac{dP_{\mathcal{D}_s}}{dP_{\boldsymbol{\nu}}}(f) dQ_{\boldsymbol{\nu}_s}(f) && \text{Log rules}
\end{aligned}$$

In the second term of the previous equation we see that we arrive at the Radon-Nikodym derivative of the true posterior with respect to the prior; assuming that a proper definition of the likelihood exists, we can use the measure theoretic Bayes rule Matthews et al., 2016:

$$\frac{dP_{\mathcal{D}_s}}{dP_{\boldsymbol{\nu}}}(f) = \frac{l_{\mathcal{D}_s}(f)}{\int l_{\mathcal{D}_s}(f) dP_{\boldsymbol{\nu}}} \tag{19}$$

where $l_{\mathcal{D}_s}(f)$ is the likelihood. We will now follow Matthews et al., 2016 and include an additional assumption. Firstly, we restrict the likelihood to depend only on the locations of the data, and define a projection function $\pi_{\mathcal{D}_s}$ which, given the function, returns the function evaluated at the input locations of $\mathcal{D}_s$. In this way we will have that

$$\frac{dP_{\mathcal{D}_s}}{dP_{\boldsymbol{\nu}}}(f) = \frac{dP_{\mathcal{D}_s}}{dP_{\boldsymbol{\nu}}}(\pi_{\mathcal{D}_s}(f)) = \frac{l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})}{\int l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s}) dP_{\boldsymbol{\nu}}} \tag{20}$$

and as a result, we end up with

$$\mathbb{D}_{\text{KL}}\left(Q||P_{\mathcal{D}}\right) = \sum_{s=1}^{S}\mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\nu}_s}||P_{\boldsymbol{\nu}}\right) - \mathbb{E}_{f\sim Q_{\boldsymbol{\nu}_s}}\left[\log l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})\right] + \underbrace{\log\int l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})dP_{\boldsymbol{\nu}}(f_{\boldsymbol{X}_s})}_{const.}$$

$$\implies \sum_{s=1}^{S}\log\int l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})dP_{\boldsymbol{\nu}}(f_{\boldsymbol{X}_s}) \geq \sum_{s=1}^{S}\mathbb{E}_{f\sim Q_{\boldsymbol{\nu}_s}}\left[\log l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})\right] - \mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\nu}_s}||P_{\boldsymbol{\nu}}\right) \tag{21}$$

C.3 `FedVEM` `HSVI` derivation

Here we show that the `FedVEM` `HSVI` approach is still maximizing a lower bound to the marginal likelihood. This can be understood by considering the `FedVEM` `FSVI` objective

$$\sum_{s=1}^{S}\log\int l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})dP_{\boldsymbol{\nu}}(f_{\boldsymbol{X}_s}) \geq \sum_{s=1}^{S}\mathbb{E}_{f\sim Q_{\boldsymbol{\nu}_s}}\left[\log l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})\right] - \mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\nu}_s}||P_{\boldsymbol{\nu}}\right) \tag{22}$$

Notice that the only term involving the prior hyper parameters $\boldsymbol{\nu}$ is the KL-divergence $\mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\nu}_s}||P_{\boldsymbol{\nu}}\right)$. Since from the data processing inequality from Proposition C.1 we know that

$$\mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\nu}_s}||P_{\boldsymbol{\nu}}\right) \leq \mathbb{D}_{\text{KL}}\left(q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)||p_{\boldsymbol{\nu}}(\boldsymbol{\theta}_s)\right) \tag{23}$$

we can show that the functional ELBO is an upper bound to the weight-space ELBO

$$\begin{aligned}\sum_{s=1}^{S}\log\int l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})dP_{\boldsymbol{\nu}}(f_{\boldsymbol{X}_s}) &\geq \sum_{s=1}^{S}\mathbb{E}_{f\sim Q_{\boldsymbol{\nu}_s}}\left[\log l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})\right] - \mathbb{D}_{\text{KL}}\left(Q_{\boldsymbol{\nu}_s}||P_{\boldsymbol{\nu}}\right)\\ &\geq \sum_{s=1}^{S}\mathbb{E}_{f\sim Q_{\boldsymbol{\nu}_s}}\left[\log l_{\mathcal{D}_s}(f_{\boldsymbol{X}_s})\right] - \mathbb{D}_{\text{KL}}\left(q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)||p_{\boldsymbol{\nu}}(\boldsymbol{\theta}_s)\right).\end{aligned} \tag{24}$$

Therefore, the weight-space ELBO optimized at the server is a proper lower bound to the log marginal-likelihood.

C.4 `FedAvg` `FSVI` derivation

We start with the standard `FSVI` ELBO from Rudner et al., 2021

$$\sum_{i=1}^{N}\mathbb{E}_{q_{\boldsymbol{\nu}}(f;\boldsymbol{\theta})}\left[\log p(\mathbf{y}_i|\mathbf{x}_i,\boldsymbol{\theta})\right] - \mathbb{D}_{\text{KL}}\left(\tilde{q}_{\boldsymbol{\nu}}\left(\tilde{f}\left(\boldsymbol{X}_I;\boldsymbol{\theta}\right)\right)\middle|\middle|\tilde{p}\left(\tilde{f}\left(\boldsymbol{X}_I;\boldsymbol{\theta}\right)\right)\right), \tag{25}$$

where $\boldsymbol{\nu}$ are the variational parameters to be optimized. Assuming that there are M datapoints in $\boldsymbol{X}_I$, Rudner et al., 2021 assume independence between those datapoints, so the KL term can be decomposed as follows

$$\mathbb{D}_{\text{KL}}\left(\tilde{q}_{\boldsymbol{\nu}}\left(\tilde{f}\left(\boldsymbol{X}_I;\boldsymbol{\theta}\right)\right)\middle|\middle|\tilde{p}\left(\tilde{f}\left(\boldsymbol{X}_I;\boldsymbol{\theta}\right)\right)\right) = \sum_{m=1}^{M}\mathbb{D}_{\text{KL}}\left(\tilde{q}_{\boldsymbol{\nu}}\left(\tilde{f}\left(\boldsymbol{x}_m;\boldsymbol{\theta}\right)\right)\middle|\middle|\tilde{p}\left(\tilde{f}\left(\boldsymbol{x}_m;\boldsymbol{\theta}\right)\right)\right). \tag{26}$$

Now assuming that each client has a subset of the inducing points $\boldsymbol{X}_I$, *i.e.*, $\boldsymbol{X}_{Is}$, of size M_s, we can rewrite the above objective as

$$\begin{aligned}&\sum_{i=1}^{N}\mathbb{E}_{q_{\boldsymbol{\nu}}(f;\boldsymbol{\theta})}\left[\log p(\mathbf{y}_i|\mathbf{x}_i,\boldsymbol{\theta})\right] - \sum_{m=1}^{M}\mathbb{D}_{\text{KL}}\left(\tilde{q}_{\boldsymbol{\nu}}\left(\tilde{f}\left(\boldsymbol{x}_m;\boldsymbol{\theta}\right)\right)\middle|\middle|\tilde{p}\left(\tilde{f}\left(\boldsymbol{x}_m;\boldsymbol{\theta}\right)\right)\right) =\\ &\sum_{s=1}^{S}\left(\sum_{i=1}^{N_s}\mathbb{E}_{q_{\boldsymbol{\nu}}(f;\boldsymbol{\theta})}\left[\log p(\mathbf{y}_i|\mathbf{x}_i,\boldsymbol{\theta})\right] - \sum_{m=1}^{M_s}\mathbb{D}_{\text{KL}}\left(\tilde{q}_{\boldsymbol{\nu}}\left(\tilde{f}\left(\boldsymbol{x}_m;\boldsymbol{\theta}\right)\right)\middle|\middle|\tilde{p}\left(\tilde{f}\left(\boldsymbol{x}_m;\boldsymbol{\theta}\right)\right)\right),\right)\end{aligned} \tag{27}$$

which is a sum of local optimization problems, with $N = \sum_{s=1}^{S} N_s$ and $M = \sum_{s=1}^{S} M_s$. Consequently, `FedAvg` can be applied in order to optimize $\boldsymbol{\nu}$.

D Weight-space Variational Inference

D.1 Variational Bayes

We start with the standard ELBO and transform it into a sum of local optimization problems:

$$L_{\mathcal{D}}(\boldsymbol{\nu}) = \mathbb{E}_{\boldsymbol{\theta}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta})}\left[\log l_{\mathcal{D}}(f;\boldsymbol{\theta})\right] - \mathbb{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\nu}}(\boldsymbol{\theta})\right) \tag{28}$$

$$= \mathbb{E}_{\boldsymbol{\theta}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta})}\left[\sum_{s=1}^{S}\log l_{\mathcal{D}_s}(f)\right] - \mathbb{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\nu}}(\boldsymbol{\theta})\right) \tag{29}$$

$$= \sum_{s=1}^{S}\left[\mathbb{E}_{\boldsymbol{\theta}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta})}\left[l_{\mathcal{D}_s}(f)\right] - \frac{1}{S}\mathbb{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\nu}}(\boldsymbol{\theta})\right)\right] = \sum_{s=1}^{S} L_{\mathcal{D}_s}(\boldsymbol{\nu}), \tag{30}$$

where

$$\nabla_{\boldsymbol{\nu}} L_{\mathcal{D}_s}(\boldsymbol{\nu}) \approx \nabla_{\boldsymbol{\nu}}\left[\mathbb{E}_{\boldsymbol{\theta}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta})}\left[\frac{|\mathcal{D}_s|}{M}\sum_{i-1}^{M}\log l_{(\boldsymbol{x}_{s,i},\boldsymbol{y}_{s,i})}(f;\boldsymbol{\theta})\right] - \frac{1}{S}\mathbb{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\nu}}(\boldsymbol{\theta})\right)\right] \tag{31}$$

$$\approx |\mathcal{D}_s|\nabla_{\boldsymbol{\nu}}\left[\mathbb{E}_{\boldsymbol{\theta}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta})}\left[\frac{1}{M}\sum_{i-1}^{M}\log l_{(\boldsymbol{x}_{s,i},\boldsymbol{y}_{s,i})}(f;\boldsymbol{\theta})\right] - \frac{1}{S\cdot|\mathcal{D}_s|}\mathbb{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\nu}}(\boldsymbol{\theta})\right)\right], \tag{32}$$

$(\boldsymbol{x}_{s,i}, \boldsymbol{y}_{s,i})$ corresponds to the i^{th} datapoint on client s and $\boldsymbol{\theta} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta})$ implicitely using the local reparameterization trick (Kingma, Salimans, et al., 2015). As is typically done in `FedAvg`, the server interprets the difference $(\boldsymbol{\nu}^t - \boldsymbol{\nu}_s^{t+1})$ between the current server-parameters $\boldsymbol{\nu}^t$ and the updated parameters by client s as a gradient to be applied by a server-side optimizer. The server is responsible for weighing the individual clients' updates by their local data-set size D_s. Both, $p_{\boldsymbol{\nu}}(\boldsymbol{\theta})$ and $q_{\boldsymbol{\phi}}(\boldsymbol{\theta})$ are parameterized as Gaussian distributions under the mean-field assumption.

D.2 Federated Variational Expectation Maximization

We start with the objective established in (Louizos et al., 2021).

$$L(\boldsymbol{\theta}, \boldsymbol{\nu}_{1:S}) = \sum_{s=1}^{S}\mathbb{E}_{\boldsymbol{\theta}_s\sim q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)}\left[\log p(\mathcal{D}_s|\boldsymbol{\theta}_s) + \log p(\boldsymbol{\theta}_s|\boldsymbol{\nu}) - \log q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)\right] \tag{33}$$

We interpret the server weights as the hyperparameters of some distribution, $\boldsymbol{\nu} = [\boldsymbol{\mu}, \boldsymbol{\sigma}]$, in this case a normal distribution. Locally, clients optimize equation (33) with respect to $\boldsymbol{\nu}_s$ while the server optimizes $\boldsymbol{\nu}$:

$$\nabla_{\boldsymbol{\mu}} L(\boldsymbol{\nu}, \boldsymbol{\nu}_{1:S}) = \sum_{s=1}^{S}\mathbb{E}_{\boldsymbol{\theta}_s\sim q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)}[\nabla_{\boldsymbol{\mu}}\log p(\boldsymbol{\theta}_s|\boldsymbol{\nu})] \tag{34}$$

$$= \sum_{s=1}^{S}\mathbb{E}_{\boldsymbol{\theta}_s\sim q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)}[-\frac{1}{\boldsymbol{\sigma}^2}(\boldsymbol{\mu} - \boldsymbol{\theta}_s)] = 0 \tag{35}$$

$$\implies \boldsymbol{\mu}^* = \frac{1}{S}\sum_{s=1}^{S}\mathbb{E}_{\boldsymbol{\theta}_s\sim q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)}[\boldsymbol{\theta}_s] = \frac{1}{S}\sum_{s=1}^{S}\boldsymbol{\mu}_s \tag{36}$$

For $\boldsymbol{\sigma}^2$ we parameterize $\boldsymbol{\sigma}^2 = \exp \boldsymbol{\alpha}$ and investigate

$$\nabla_{\boldsymbol{\alpha}} L(\boldsymbol{\theta}, \boldsymbol{\nu}_{1:S}) = \boldsymbol{\sigma}^2 \nabla_{\boldsymbol{\sigma}^2} L(\boldsymbol{w}) \tag{37}$$

$$= \boldsymbol{\sigma}^2 \sum_{s=1}^{S} \mathbb{E}_{\boldsymbol{\theta}_s \sim q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)} [\nabla_{\boldsymbol{\sigma}^2} \log p(\boldsymbol{\theta}_s | \boldsymbol{\nu})] \tag{38}$$

$$= \boldsymbol{\sigma}^2 \sum_{s=1}^{S} \mathbb{E}_{\boldsymbol{\theta}_s \sim q_{\boldsymbol{\nu}_s}(\boldsymbol{\theta}_s)} \left[-\frac{1}{2\boldsymbol{\sigma}^2} + \frac{1}{2{\boldsymbol{\sigma}^2}^2} (\boldsymbol{\mu} - \boldsymbol{\theta}_s)^2 \right] \tag{39}$$

$$= -\frac{S}{2} + \frac{1}{2\boldsymbol{\sigma}^2} \sum_{s=1}^{S} [(\boldsymbol{\mu} - \boldsymbol{\mu}_s)^2 + \boldsymbol{\sigma}_s^2] = 0 \tag{40}$$

$$\implies \boldsymbol{\sigma}^{2*} = \frac{1}{S} \sum_{s=1}^{S} [(\boldsymbol{\mu} - \boldsymbol{\mu}_s)^2 + \boldsymbol{\sigma}_s^2] \tag{41}$$

$$\implies \boldsymbol{\alpha}^* = \log \left(\frac{1}{S} \sum_{s=1}^{S} [(\boldsymbol{\mu} - \boldsymbol{\mu}_s)^2 + \boldsymbol{\sigma}_s^2] \right) \tag{42}$$

We interpret $\boldsymbol{\mu}^t - \boldsymbol{\mu}^{*t}$ as well as $\boldsymbol{\alpha}^t - \boldsymbol{\alpha}^{*t+1}$ as gradients in the usual way to be applied by a server-side optimizer.

E Experimental details

E.1 Hyperparameter tuning

Weight-space methods (Reddi et al., 2021) showed the importance of jointly tuning the local learning rate η_s, the server learning rate η and optionally τ when using more advanced server-side optimizers than SGD. Consequently, we tune these three parameters extensively on `FedAvg WSVI` for our MNIST experiments, as detailed in Table 2. For this initial sweep, we initialize $\sigma^2_{init} = 1e^{-5}$ and train for only 300 communication rounds. This low value for initializing the σ^2 of $q_\phi(\theta)$ causes the model to have very low sampling noise in its gradients and ensures faster convergence. Subsequently, we tune σ^2_{init} and train for 500 rounds in order to allow enough time for the KL-term in the ELBO objective to decrease. We found the hyperparameters for η, η_s and τ to generalize well to `FedVEM WSVI` and only tune σ^2_{init} as described for `FedAvg WSVI`. For `PVI`, the dampening factor ρ plays a role similar to the server-side learning rate η. We first tune η_s and σ^2_{init} jointly, keeping $\rho = 0.96$ and training for 500 rounds. We observe that the training loss tends to increase after some point during training, which we mitigate by setting a higher initial value of $\rho = 0.98$ annealing ρ during training from 0.98 to 0.999 following a cosine annealing schedule.

For the regression case, we tune hyper parameters in an unstructured way and find the use of Adam at the server important. For both, `FedVEM WSVI` and `FedAvg WSVI`, we set $\eta = 0.01$ and keep τ at the default value. Curiously, we found that for the local optimizer, `FedVEM WSVI` requires Adam ($\eta = 0.001$) to function well, whereas `FedAvg WSVI` requires SGD ($\eta = 0.001$). For `PVI`, we use Adam ($\eta = 0.0001$) locally and set $\rho = 0$.

Function-space methods For the function-space VI we use the same hyperparameters as the ones from the weight-space VI methods. We also introduced one more hyperparameter, a weighting factor β in front of the function-space KL-divergence. This was the only extra hyperparameter that we tuned. For the inducing point strategy at each client we sampled a random datapoint from the local dataset and added independent uniform noise on each (normalized) pixel by sampling from $U[-1/255, 1/255]$ on each iteration of local training.

E.2 Experimental details

Federated MNIST The MNIST dataset is not naturally split into clients. In order to have an artificial testbed for federate algorithms using MNIST, we split the training-set into $S = 100$ clients in a non-i.i.d. fashion according to the labels. Each client's label distribution is drawn from Dir($\alpha =$

Table 2: Hyperparameter ranges for MNIST

Method	Parameter	Ranges	Chosen
PVI	η_s	$\{10^{-3}, 10^{-2.5}, 10^{-2}, \dots, 10^{-1}\}$	10^{-3}
	σ^2_{init}	$\{10^{-4.5}, 10^{-4}, 10^{-3.5}\}$	10^{-4}
FedAvg WSVI	η_s	$\{10^{-3}, 10^{-2.5}, 10^{-2}, \dots, 10^{0}\}$	10^{-1}
	η	$\{10^{-3}, 10^{-2.5}, 10^{-2}, \dots, 10^{-1}\}$	$10^{-2.5}$
	τ	$\{10^{-5}, 10^{-4.5}, 10^{-3}, \dots, 10^{-1}\}$	10^{-4}
	σ^2_{init}	$\{10^{-5}, 10^{-4.5}, 10^{-4}, \dots, 10^{-2.5}\}$	10^{-3}
FedVEM WSVI	σ^2_{init}	$\{10^{-5}, 10^{-4.5}, 10^{-4}, \dots, 10^{-2.5}\}$	10^{-3}
FedAvg FSVI	β	$\{1, 0.1, 0.05, 0.01, 0.005, 0.001\}$	0.005
FedVEM FSVI	β	$\{1, 0.1, 0.05, 0.01, 0.005, 0.001\}$	0.005

1.0), resulting in heavily skewed distributions. All models are trained for 3000 communication rounds.

Regression Our toy regression data-set is created by sampling 200 data-points $\boldsymbol{x}'$ uniformly in $[0, 1)$. Subsequently, these data-points are scaled to $\boldsymbol{x} = 5 \cdot (\boldsymbol{x}' \cdot (0.82 - 0.1) + 0.1)$. We sample 200 random noise elements $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, 0.03\boldsymbol{I})$ and compute

$$\boldsymbol{y} = \sin\left(\frac{-3\pi}{5}(\boldsymbol{x} + \boldsymbol{\epsilon}) + 0.5\right) + \cos\left(\frac{-6\pi}{5}(\boldsymbol{x} + \boldsymbol{\epsilon})\right) + 8\boldsymbol{\epsilon}. \tag{43}$$

We randomly select 160 input-output pairs (x, y) to keep as training-set and use the remaining 40 for validation. For creating a non-i.i.d. federated split of the data, we sort the training-set according to x and assign one half of the data-points to each of the 2 clients. All models are trained for 3000 communication rounds.

Models For MNIST classification, we use the LeNet5-architecture LeCun et al., 1998 without dropout or weight-decay. The regression experiments are performed with a 2-layer MLP of 100 hyperbolic tangent units each.

Optimizers For all our methods, we use Adam Kingma and Ba, 2017 as server-side optimizer and SGD as client-side optimizer. The exception is with PVI, which prescribes its own server-side update rule and introduces dampening as described in the previous section. We found using Adam as client-side optimizer crucial for good performance with PVI.

Seeds We repeat the MNIST experiments with different seeds. Standard-error in Tables 1 and 4 are based on 8 seeds for FedVEM WSVI, 7 seeds for FedAvg WSVI and 8 seeds for PVI. For FedAvg FSVI we evaluate on 2 seeds and on 3 seeds for FedVEM HSVI, due to time constraints.

F Additional experiments

F.1 Toy regression metrics

For a quantitative evaluation of model fit and out of distribution performance, Table 3 contains negative log-likelihood, expected calibration error as defined in Levi et al., 2020 a well as the area under the receiver operating curve (OOD-AUROC) based on the standard-deviation for individual data-points based on 100 samples from the posterior. OOD is defined as the union of two ranges in the input space, one with lower values than then in-distribution data and one with higher values. Specifically, the OOD ranges to the left and to the right as wide as the range of in-distribution data. We sample 1000 points from this range to evaluate OOD-AUROC.

FedVEM HSVI offers the best-calibrated predictions, although it sacrifices a little in likelihood compared to FedVEM WSVI which also offers more convincing OOD detection performance. Although

Table 3: Metrics for toy regression task

Method	NLL ↓	ECE ↓	OOD-AUROC ↑
PVI	0.3442	45.26	21.61
FedAvg WSVI	1.2293	53.08	10.0
FedVEM WSVI	0.2192	53.54	54.47
FedAvg FSVI	0.4218	37.28	56.51
FedVEM HSVI	0.2994	35.52	36.52

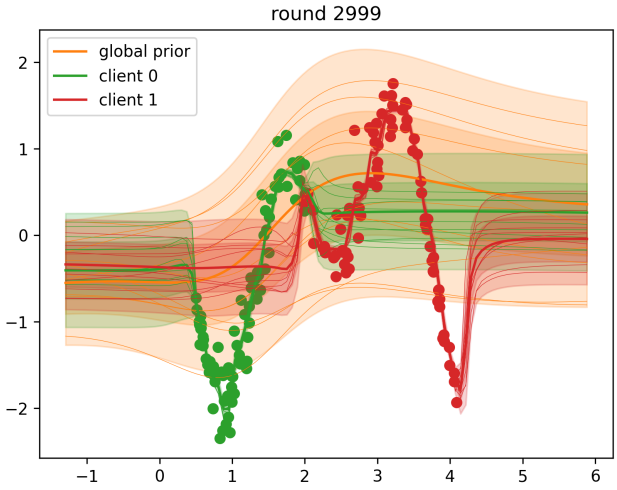


Figure 2: Plot of the fit in the main paper for FedVEM FSVI, with the local client functions plotted as well

FedAvg FSVI appears to be well calibrated and useful for OOD detection, the figures in the main text reveal that the learned functions miss the details of the data-set.

F.2 OOD-AUROC on MNIST

In the main-text, we report the average OOD-AUROC across Fashion-MNIST, Gaussian noise and uniform noise. Here we take these apart and report the OOD-AUROC on individual out of distribution datasets in Table 4.

Table 4: OOO-AUROC on different out of distribution datasets

Method	Fashion MNIST	Uniform	Gaussian
PVI	$99.30_{\pm 0.05}$	$100.0_{\pm 0}$	$98.67_{\pm 0.05}$
FedAvg WSVI	$99.21_{\pm 0.13}$	$100.0_{\pm 0}$	$99.17_{\pm 0.13}$
FedVEM WSVI	$95.33_{\pm 0.5}$	$99.88_{\pm 0.03}$	$99.95_{\pm 0.02}$
FedAvg FSVI	$99.60_{\pm 0.18}$	$99.99_{\pm 0.01}$	$99.20_{\pm 0.55}$
FedVEM HSVI	$98.74_{\pm 0.17}$	$99.95_{\pm 0.04}$	$98.29_{\pm 0.57}$