EXPRESSIVE YET TRACTABLE BAYESIAN DEEP LEARNING VIA SUBNETWORK INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

The Bayesian paradigm has the potential to solve some of the core issues in modern deep learning, such as poor calibration, data inefficiency, and catastrophic forgetting. However, scaling Bayesian inference to the high-dimensional parameter spaces of deep neural networks requires restrictive approximations. In this paper, we propose performing inference over only a small subset of the model parameters while keeping all others as point estimates. This enables us to use expressive posterior approximations that would otherwise be intractable for the full model. In particular, we develop a practical and scalable Bayesian deep learning method that first trains a point estimate, and then infers a full covariance Gaussian posterior approximation over a subnetwork. We propose a subnetwork selection procedure which aims to optimally preserve posterior uncertainty. We empirically demonstrate the effectiveness of our approach compared to point-estimated networks and methods that use less expressive posterior approximations over the full network.

1 INTRODUCTION

Deep neural networks (DNNs) still suffer from critical shortcomings that make them unfit for important applications. For instance, DNNs tend to be *poorly calibrated and overconfident* in their predictions, especially when there is a shift in the train and test distributions (Nguyen et al., 2015; Guo et al., 2017). To reliably inform decision making, DNNs must be able to robustly quantify the *uncertainty* in their predictions, which is particularly important in safety-critical areas such as healthcare or autonomous driving (Amodei et al., 2016).

Bayesian modeling (Gal, 2016; Ghahramani, 2015) presents a principled way to capture predictive uncertainty via the posterior distribution over model parameters. Unfortunately, due to their nonlinearities, exact posterior inference is intractable in NNs. Despite recent successes in the field of Bayesian deep learning (Gal & Ghahramani, 2016; Blundell et al., 2015; Osawa et al., 2019; Maddox et al., 2019; Dusenberry et al., 2020), existing methods are only made scalable to modern DNNs with large numbers of parameters by invoking unrealistic assumptions. This severely limits the expressiveness of the inferred posterior and thus deteriorates the quality of the uncertainty estimates (Ovadia et al., 2019; Fort et al., 2019; Ashukha et al., 2020; Foong et al., 2019a).

Due to the heavy overparameterization of DNNs, their accuracy is well-preserved by a small subnetwork (Cheng et al., 2017). Additionally, recent work by Izmailov et al. (2019) has shown how performing inference over a low dimensional subspace of the weights can result in accurate uncertainty quantification. These observations prompt the following question for a DNN's uncertainty: *Can a full DNN's model uncertainty be well-preserved by a small subnetwork's model uncertainty?* We answer this question in the affirmative. We show both theoretically and empirically that the full network posterior is well represented by a subnetwork's posterior. In turn, we can apply more expensive, but more faithful, posterior approximations to just that subnetwork to achieve better uncertainty quantification than if we apply cheaper, but more crude, approximations to the full network.

The contributions of this paper are as follows:

1. We propose a new Bayesian deep learning approach that performs Bayesian inference over only a small subset of the model weights and keeps all other weights deterministic. This allows us to use expressive posterior approximations that are typically intractable in DNNs.

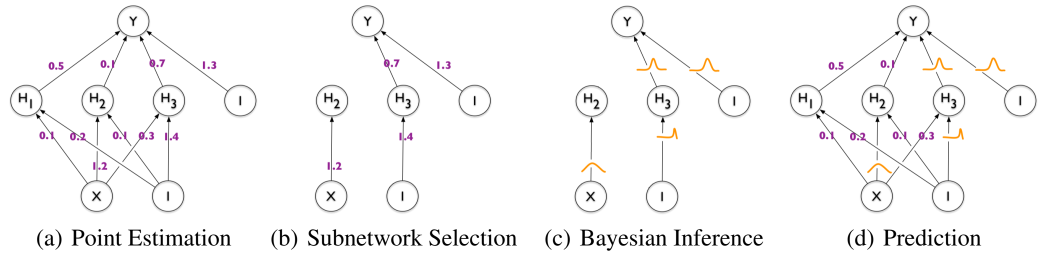

Figure 1: Schematic illustration of our proposed approach. (a) We train a neural network using standard techniques to obtain a point estimate of the weights. (b) We identify a small subset of the weights. (c) We estimate a posterior distribution over the selected subnetwork via Bayesian inference techniques. (d) We make predictions using the full network of mixed Bayesian/deterministic weights.

2. As a concrete instantiation of this framework, we develop a practical and scalable Bayesian deep learning method that uses the linearized Laplace approximation to infer a full-covariance Gaussian posterior over a subnetwork within a point-estimated neural network.
3. We theoretically characterize the discrepancy between the posterior distributions over a subnetwork and the full network (in terms of their Wasserstein distance) in the linearized model, and derive an optimal strategy to select a subnetwork that minimizes this discrepancy.
4. We empirically show on various benchmarks that our method compares favourably against point-estimated networks and other Bayesian deep learning methods, experimentally confirming that expressive subnetwork inference is superior to crude inference over full networks.

2 Subnetwork Posterior Approximation

Bayesian neural networks (BNNs) aim to capture *model uncertainty*, i.e., uncertainty about the choice of weights $\boldsymbol{W}$ due to multiple plausible explanations of the training data $\{\boldsymbol{y}, \boldsymbol{X}\}$, where $\boldsymbol{y}$ is the dependent variable (e.g. classification label) and $\boldsymbol{X}$ is the feature matrix. A prior distribution $p(\mathbf{W})$ is specified over the BNN's weights, and we wish to infer their full *posterior distribution*

$$p(\mathbf{W}|\boldsymbol{y}, \boldsymbol{X}) \quad \propto \quad p(\boldsymbol{y}|\boldsymbol{X}, \mathbf{W})\, p(\mathbf{W})\,. \tag{1}$$

To make predictions, we then estimate the *posterior predictive* distribution that averages the network's predictions across all possible settings of the weights, weighted by their posterior probability, i.e.

$$p(\mathbf{y}^*|\boldsymbol{X}^*, \boldsymbol{y}, \boldsymbol{X}) = \int_{\mathbf{W}} p(\mathbf{y}^*|\boldsymbol{X}^*, \mathbf{W}) p(\mathbf{W}|\boldsymbol{y}, \boldsymbol{X}) d\mathbf{W}\,. \tag{2}$$

Unfortunately, due to the size of modern deep neural networks, it is not only intractable to infer the exact posterior distribution $p(\mathbf{W}|\boldsymbol{y}, \boldsymbol{X})$ in Eq. (1), but it is even challenging to properly approximate it. As a consequence, extreme posterior approximations such as complete factorization are commonly employed (Blundell et al., 2015; Hernández-Lobato & Adams, 2015b; Kingma et al., 2015; Khan et al., 2018; Osawa et al., 2019), i.e. $p(\mathbf{W}|\boldsymbol{y}, \boldsymbol{X}) \approx \prod_{d=1}^{D} q(\mathrm{w}_d)$ where w_d denotes the d-th weight in the D-dimensional neural network weight vector $\mathbf{W} \in \mathbb{R}^D$ (which we for simplicity obtain by concatenating and flattening the weight matrices of all network layers). Clearly, this is a very 'wishful' approximation and known to suffer from severe pathologies (Foong et al., 2019a;b).

In this work, we question the implicit assumption that a good posterior approximation needs to include *all* BNN parameters. Instead, we aim to perform inference only over a *small subset* of the weights. This approach is well-motivated for at least two reasons:

1. **Overparameterization:** Maddox et al. (2020) have shown that, in the neighborhood of local optima, there are many directions that leave the NN's predictions unchanged. Moreover, NNs can be heavily pruned without sacrificing test-set accuracy (Frankle & Carbin, 2019). Thus, the majority of a NN's predictive power is isolated to a small subnetwork.

2. **Inference over submodels:** Previous work[1] has provided evidence that inference can be effective even when not done on the full parameter space. Izmailov et al. (2019) performed inference over a low-dimensional projection of the weights. Neural-linear models, which give a Bayesian treatment to only the last layer of a DNN, have shown to be competitive with full-network approaches (Riquelme et al., 2018; Ober & Rasmussen, 2019).

We thus combine these ideas, making the following two-step approximation of the posterior in Eq. (1):

$$p(\mathbf{W}|\boldsymbol{y}, \boldsymbol{X}) \;\approx\; p(\mathbf{W}_S|\boldsymbol{y}, \boldsymbol{X}) \prod_r \delta(\mathrm{w}_r - \mathrm{w}_r^*) \;\approx\; q(\mathbf{W}_S) \prod_r \delta(\mathrm{w}_r - \mathrm{w}_r^*) \,. \tag{3}$$

The first approximation decomposes the full neural network posterior $p(\mathbf{W}|\boldsymbol{y}, \boldsymbol{X})$ into a posterior $p(\mathbf{W}_S|\boldsymbol{y}, \boldsymbol{X})$ over the subnetwork $\mathbf{W}_S$ and delta functions $\delta(\mathrm{w}_r - \mathrm{w}_r^*)$ over all remaining weights $\{\mathrm{w}_r\}_r$, keeping them at fixed values $\mathrm{w}_r^* \in \mathbb{R}$. This can be viewed as *pruning the variances* of the weights $\{\mathrm{w}_r\}_r$ to zero, which is in contrast to ordinary weight pruning methods that set the weights $\{\mathrm{w}_r\}_r$ themselves to zero. The second approximation is a result of posterior inference over the subnetwork still being intractable. In turn, we introduce the variational distribution $q(\mathbf{W}_S)$. Yet, as the subnetwork is much smaller than the full network, we can afford to make $q(\mathbf{W}_S)$ expressive and able to capture rich dependencies across the weights within the subnetwork.

3 Inference via Laplace Approximation

To obtain a method that is as practical as possible, we propose to use inference techniques that can estimate a posterior distribution *post-hoc* from a point-estimated network. The *Laplace approximation* (MacKay, 1992) is well-suited to this task as it derives the approximate posterior from the local optimization landscape. Other related inference procedures such as SWAG (Maddox et al., 2019) could also be used, but we focus on Laplace due to it being a well-studied, fundamental technique.

Step #1: Point Estimation. The first step of the procedure is to train a neural network to obtain a point estimate of the weights, denoted $\boldsymbol{W}_{MAP}$. This estimate should respect the Bayesian model given in Eq. (1), and therefore we optimize the *maximum a-posteriori* (MAP) objective:

$$\boldsymbol{W}_{MAP} = \arg\max_{\boldsymbol{W}} \left[\log p(\mathbf{y}|\boldsymbol{X}, \boldsymbol{W}) + \log p(\boldsymbol{W})\right] \,. \tag{4}$$

This can be done using standard stochastic gradient-based optimization methods commonly-used in modern deep learning (Goodfellow et al., 2016). This step is illustrated in Fig. 1 (a).

Step #2: Subnetwork Selection. The second step is to identify a small subnetwork $\boldsymbol{W}_S$. Ideally, we would like to identify the subnetwork whose posterior is 'closest' to the full-network posterior. We formalize this argument in Section 4 and describe a principled strategy that, under certain conditions, minimizes the 2-Wasserstein distance between the sub- and full-network posteriors. All other weights not belonging to that subnetwork are then assigned fixed values: the MAP estimates obtained in Step #1. We emphasize that this whole procedure is a perfectly valid mixed inference strategy: full Laplace inference over the subnetwork and MAP inference over the remaining weights. See Fig. 1 (b) for an illustration of this step.

Step #3: Variational Inference. Given the subnetwork point estimate $\boldsymbol{W}_{MAP}^S$, we use the Laplace approximation to infer a full-covariance Gaussian posterior over the subnetwork $\mathbf{W}_S$:

$$p(\mathbf{W}_S|\boldsymbol{y}, \boldsymbol{X}) \;\approx\; q(\mathbf{W}_S) \;=\; \mathcal{N}\left(\mathbf{W}_S; \boldsymbol{W}_{MAP}^S, H^{-1}\right) \tag{5}$$

where the posterior covariance matrix $H^{-1} \in \mathbb{R}^{D \times D}$ corresponds to the inverse of the average Hessian of the negative log posterior, i.e. $H = N\mathbb{E}\left[-\partial^2 \log p(\mathbf{y}|\boldsymbol{X}, \boldsymbol{W})/\partial \boldsymbol{W}^2\right] + \lambda \boldsymbol{I}$, where the expectation is w.r.t. the data generating distribution, and where λ is the precision of the zero-mean factorized Gaussian prior $p(\mathbf{W}) = \mathcal{N}(\mathbf{W}; \mathbf{0}, \lambda^{-1}\boldsymbol{I})$. In practice, we will approximate the Hessian H with the *generalized Gauss-Newton (GGN) matrix* $\widetilde{H}$ (Schraudolph, 2002), i.e.

$$\widetilde{H} = \sum_{n=1}^{N} \boldsymbol{J}_n^\top \boldsymbol{H}_n \boldsymbol{J}_n + \lambda \boldsymbol{I}, \;\; \text{with} \;\; \boldsymbol{J}_n = \frac{\partial \boldsymbol{f}(\boldsymbol{x}_n, \boldsymbol{W})}{\partial \boldsymbol{W}} \;\; \text{and} \;\; \boldsymbol{H}_n = \frac{\partial^2 L(\boldsymbol{y}_n, \boldsymbol{f}(\boldsymbol{x}_n, \boldsymbol{W}))}{\partial^2 \boldsymbol{f}(\boldsymbol{x}_n, \boldsymbol{W})} \tag{6}$$

[1] See Section 6 for a more thorough discussion of related work.

where $\boldsymbol{J}_n \in \mathbb{R}^{O \times D}$ is the Jacobian of the neural network features $\boldsymbol{f}(\boldsymbol{x}_n, \boldsymbol{W}) \in \mathbb{R}^O$ w.r.t. the weights $\boldsymbol{W}$, and $\boldsymbol{H}_n \in \mathbb{R}^{O \times O}$ is the Hessian of the loss $L(\boldsymbol{y}_n, \boldsymbol{f}(\boldsymbol{x}_n, \boldsymbol{W}))$ w.r.t. the features $\boldsymbol{f}(\boldsymbol{x}_n, \boldsymbol{W})$.The GGN $\widetilde{H}$ has clear practical advantages over the Hessian H; see Martens & Sutskever (2011) and Martens (2016). Using the Laplace approximation with the GGN can be viewed as an implicit *local linearization* of the underlying neural network $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{W})$ at its MAP estimate $\boldsymbol{W}_{MAP}$, i.e.

$$\boldsymbol{f}_{lin}^{MAP}(\boldsymbol{x}, \boldsymbol{W}) = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{W}_{MAP}) + \boldsymbol{J}_{\boldsymbol{W}_{MAP}}(\boldsymbol{x})(\boldsymbol{W} - \boldsymbol{W}_{MAP}) \tag{7}$$

where $\boldsymbol{J}_{\boldsymbol{W}_{MAP}}(\boldsymbol{x}) = \partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{W}_{MAP}) / \partial \boldsymbol{W}_{MAP}$ (Immer et al., 2020). The GGN approximation thus locally turns the underlying probabilistic model from a Bayesian neural network into a (generalized) linear model (Immer et al., 2020), which is a particularly useful property that will allow us to derive a principled subnetwork selection strategy in Section 4. This step is illustrated in Fig. 1 (c).

Step #4: Prediction. Given the linearized Laplace approximation over the subnetwork $\mathbf{W}_S$ in Eqs. (5) and (6), i.e. $q(\mathbf{W}_S) = \mathcal{N}(\mathbf{W}_S; \boldsymbol{W}_{MAP}^S, \widetilde{H}^{-1})$, we can then compute the posterior predictive distribution. While traditionally, one would compute the predictive distribution using the original Bayesian neural network likelihood, i.e. $p(\mathbf{y}|\boldsymbol{X}, \mathbf{W}) = p(\mathbf{y}|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{W}))$, Immer et al. (2020) recently suggested that since inference was (implicitly) done in the GGN-linearized model, it is more principled to instead predict using the linearized likelihood, i.e. $p(\mathbf{y}|\boldsymbol{X}, \mathbf{W}) = p(\mathbf{y}|\boldsymbol{f}_{lin}^{MAP}(\boldsymbol{x}, \boldsymbol{W}))$ (see Eq. (7)), providing a formal justification for the empirical superiority of this approach observed previously (Lawrence, 2001; Foong et al., 2019b). We thus obtain the *linearized predictive distribution*

$$p(\mathbf{y}^*|\boldsymbol{X}^*, \boldsymbol{y}, \boldsymbol{X}) \approx \int_{\mathbf{W}} p(\mathbf{y}^*|\boldsymbol{f}_{lin}^{MAP}(\boldsymbol{X}^*, \boldsymbol{W}))\, \mathcal{N}(\mathbf{W}_S; \boldsymbol{W}_{MAP}^S, \widetilde{H}^{-1}) \prod_r \delta(\mathrm{w}_r - \mathrm{w}_r^*)\, d\mathbf{W}\,. \tag{8}$$

There are two ways to compute Eq. (8): Firstly, via a Monte Carlo approximation $p(\mathbf{y}^*|\boldsymbol{X}^*, \boldsymbol{y}, \boldsymbol{X}) \simeq \frac{1}{M}\sum_{m=1}^{M} p(\mathbf{y}^*|\boldsymbol{f}_{lin}^{MAP}(\boldsymbol{X}^*, \widetilde{\boldsymbol{W}}_m))$ by sampling $\widetilde{\boldsymbol{W}}_m$ from $\mathcal{N}(\boldsymbol{W}_{MAP}^S, \widetilde{H}^{-1})$ and $\prod_r \delta(\mathrm{w}_r - \mathrm{w}_r^*)$, the latter of which is trivial. Secondly, due to linearity of $p(\mathbf{y}^*|\boldsymbol{f}_{lin}^{MAP}(\boldsymbol{X}^*, \boldsymbol{W}))$, there are closed-form expressions which are exact for Gaussian likelihoods (i.e. regression) and approximate for categorical ones (i.e. classification) (Bishop, 2006; Gibbs, 1998). This step is illustrated in Fig. 1 (d).

4 Posterior Gap and Optimal Subnetwork for Linear(ized) Models

We next analyze the proposed procedure in generalised linear models. In addition to having a closed-form MAP estimate, the Laplace approximation exactly corresponds to the true posterior (for a Gaussian prior), allowing us to observe how the posterior distribution over the subnetwork relates to that over the full model. We write the Bayesian regression model for response y and covariate $\boldsymbol{x}$ as

$$y_n = \mathbf{w}^T \boldsymbol{x}_n + \epsilon_n, \;\; \epsilon_n \sim N(0, \sigma_0^2), \quad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Lambda}_0^{-1}) \tag{9}$$

where σ_0^2 denotes the noise variance. We now analyze the following procedure, as described in Section 3 for neural network models:

1. Obtain a MAP estimate of the model weights[2]: $\mathbf{w}_{MAP} = (\boldsymbol{X}^T\boldsymbol{X} + \sigma_0^2\mathbf{\Lambda}_0)^{-1}\boldsymbol{X}^T\boldsymbol{y}$.
2. Select a subset of S model weights via a (one-shot) procedure of choice, yielding a binary vector $\boldsymbol{m} \in \mathbb{R}^D$ where $m_d = 1$ if the d-th weight is part of the subset, and $m_d = 0$ otherwise. For convenience, we will later use the binary mask matrix $\boldsymbol{M}_S = \boldsymbol{m}\boldsymbol{m}^\top \in \mathbb{R}^{D \times D}$ which contains ones in the rows/columns corresponding to the S weights in the subnetwork, and zeros otherwise.
3. Compute the weight posterior over the chosen subnetwork via a linearized Laplace approximation:

   $$p_S(\mathbf{w}|\boldsymbol{y}, \boldsymbol{X}) = \mathcal{N}(\mathbf{w}; \mathbf{w}_{MAP}, \boldsymbol{M}_S \odot \widetilde{H}^{-1})\,. \tag{10}$$

   Note that a) the mean of the subnetwork posterior in Eq. (10) is equal to that of the full posterior, and b) the weights not belonging to the subnetwork are deterministic, so their corresponding (co-)variances are set to zero via the element-wise product $\boldsymbol{M}_S \odot \widetilde{H}^{-1}$, yielding a $D \times D$ matrix equal to the full covariance matrix $\widetilde{H}^{-1}$ in the rows/columns of the S weights in the subnetwork, and zero in the rows/columns of all other $D - S$ weights.

[2] In the over-parameterized regime, we use the solution: $\mathbf{w}_{MAP} = \boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{X}^T + \sigma_0^2\lambda\mathbb{I}_{N\times N})^{-1}\boldsymbol{y}$.

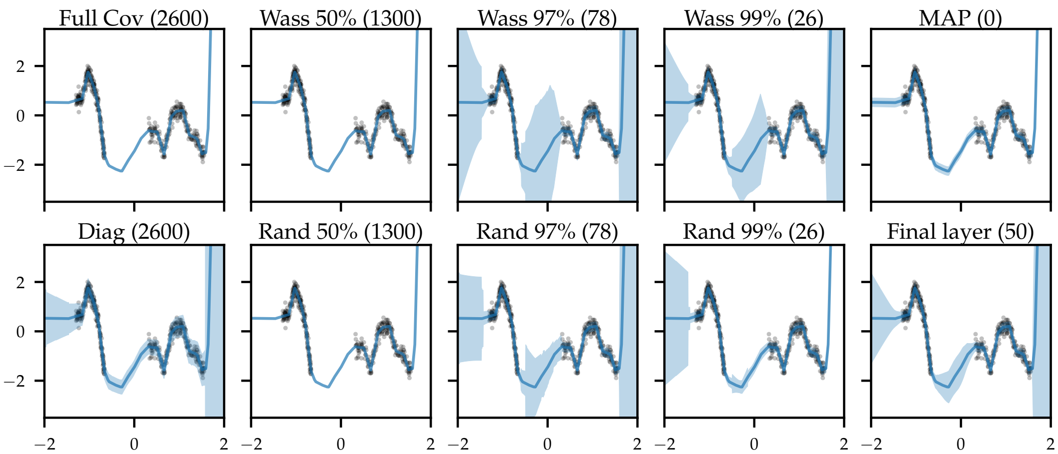

Figure 2: Predictive distributions (mean $\pm$ std) for 1D regression. The numbers in brackets denote the number of parameters over which inference was done (out of 2600 in total). Subnetwork inference via Wasserstein pruning maintains richer predictive uncertainties at smaller parameter counts.

We consider what we term the *posterior gap*—the Wasserstein distance[3] (in particular the squared 2-Wasserstein distance) between the posterior distribution over the full network and the posterior distribution over the subnetwork. The proofs for all results below will be presented in Appendix A.

Proposition 1 (Posterior Gap)**.** *For a subnetwork of size $S < D$, the Wasserstein gap between the full posterior $p(\mathbf{w}|\boldsymbol{y}, \boldsymbol{X})$ and the subnetwork posterior $p_{\mathrm{S}}(\mathbf{w}|\boldsymbol{y}, \boldsymbol{X})$ in Eq. (10) is:*

$$W\left[p(\mathbf{w}|\boldsymbol{y}, \boldsymbol{X}) \mid\mid p_{\mathrm{S}}(\mathbf{w}|\boldsymbol{y}, \boldsymbol{X})\right] = \textstyle\sum_{d=1}^{D} \left(1 + m_{dd}\right) \sigma_d^2 - \mathrm{trace}(2(\widetilde{H}^{-1}(\boldsymbol{M}_S \odot \widetilde{H}^{-1}))^{1/2}) \,. \quad (11)$$

The optimal subnetwork should then minimize the posterior gap in Eq. (11). However, for full covariance matrices $\widetilde{H}^{-1}$ and a large number of weights D, this will generally be infeasible as Eq. (11) depends on *all* entries of the $D \times D$-matrix $\widetilde{H}^{-1}$, which is intractable to compute/store. To derive a practical pruning strategy, we assume the covariance matrix to be diagonal.

Corollary 1.1 (Optimality of Maximum Variance Subnetwork Selection under Decorrelation)**.** *For a generalized linear model with posterior covariance matrix $\widetilde{H}^{-1} = diag(\sigma_1^2, \ldots, \sigma_D^2)$, the optimal subnetwork under the Wasserstein gap is comprised of the S weights with the largest variances σ_d^2.*

Finally, since a GGN-linearized neural network as in Eq. (7) formally is a generalized linear model with feature matrix $\boldsymbol{J}_{\boldsymbol{W}_{MAP}}(\boldsymbol{x}) = \partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{W}_{MAP})/\partial \boldsymbol{W}_{MAP}$ (Khan et al., 2019), Corollary 1.1 implies that the optimal subnetwork comprises of the weights with the largest variances (assuming decorrelation). In practice, even just computing the diagonal of the covariance matrix is challenging, so we use a diagonal Laplace approximation which instead computes the inverse of the diagonal of the GGN (see e.g. Ritter et al. (2018)). Note that we only have to make the decorrelation assumption for the purposes of subnetwork selection – when doing posterior inference over the selected subnetwork, *we estimate a full covariance matrix* for maximal expressiveness, as described in Step #3 in Section 3.

5 Empirical Analysis

We empirically assess the effectiveness of subnetwork inference compared to mean field inference over full networks and other uncertainty quantification methods. We consider three tasks: 1) small-scale 1D regression, 2) medium-scale tabular regression, and 3) large-scale image classification.

5.1 Does subnetwork inference retain predictive posterior uncertainty?

We assess how the predictive distribution of a full-covariance Gaussian posterior over a selected subnetwork qualitatively compares to that obtained from 1) a full-covariance Gaussian over the

[3] We use the Wasserstein distance instead of the more common Kullback–Leibler divergence because the Wasserstein is well-defined for degenerate distributions and is an actual distance metric (i.e. symmetric).

full network (Full Cov), 2) a *factorised* Gaussian posterior over the full network (Diag), 3) a full-covariance Gaussian over only the (*Final layer*) of the network (Kristiadi et al., 2020), and 4) a point estimate (MAP). For subnetwork inference, we consider both Wasserstein (Wass) (as described in Section 4) and uniform random pruning (Rand) for 50%, 97% and 99% of model parameters. Our NN consists of 2 ReLU hidden layers with 50 hidden units each. We employ a homoscedastic Gaussian likelihood function where the noise variance is optimised with maximum likelihood. We use GGN Laplace inference over network weights (not biases) in combination with the linearized predictive distribution in Eq. (8). Thus all approaches considered share their predictive mean, allowing us to better compare their uncertainty estimates. All approaches share a single prior precision λ=3.

We use a synthetic 1D regression task with two separated clusters of inputs (Antorán et al., 2020), allowing us to probe for 'in-between' uncertainty (Foong et al., 2019b). Results are shown in Fig. 2. Subnetwork inference preserves more of the uncertainty of full network inference than diagonal Gaussian or final layer inference while doing inference over fewer weights. Capturing weight correlations allows subnetwork inference to present uncertainty in between clusters of data. This is true for both random and Wasserstein pruning. However, the latter preserves more uncertainty as we prune larger amounts of weights. These results suggest **expressive inference over a carefully selected subnetwork retains more uncertainty than crude approximations applied to full networks**.

5.2 Subnetwork inference on big models vs full inference over small models

We might also ask: "why should we use subnetwork inference when we can just perform full network inference in a smaller model?" We explore this by considering 4 fully connected models of increasing size. These have numbers of hidden layers h_d=$\{1, 2\}$ and hidden layer widths w_d=$\{50, 100\}$. For a dataset with input dimension i_d, the number of weights is given by D=$(i_d$+$1)w_d$+$(h_d$−$1)w_d^2$. Our 2 hidden layer, 100 hidden unit models have a weight count of the order $1e4$. Full covariance inference in these models borders the limit of computational tractability on commercial hardware. We first obtain a MAP estimate of each model's weights and our homoscedastic likelihood function's noise variance. We then perform full network GGN Laplace inference for each model. We also prune every network's variances to the size of every smaller network under consideration using our proposed Wassertein rule. In all cases, we employ the linearised predictive model. Consequently, networks with the same number of weights make the same mean predictions. Increasing the number of weight variances we consider will only increase predictive uncertainty.

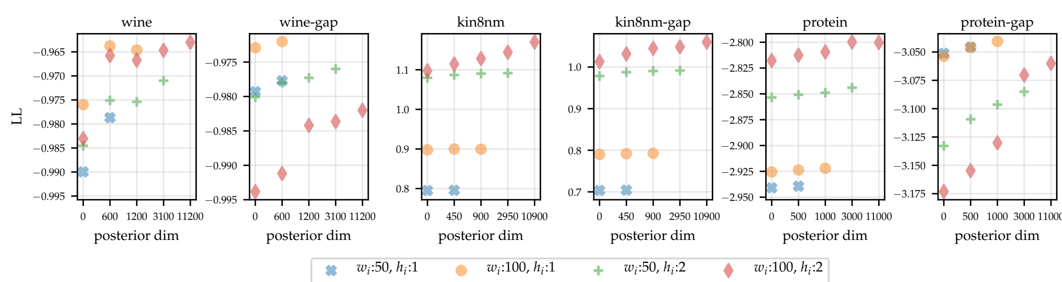

Figure 3: Mean LogLikelihood values obtained on UCI datasets across all splits. Different markers indicate models with different numbers of weights. The horizontal axis indicates the number of weights over which full covariance inference is performed. 0 corresponds to MAP parameter estimation and the rightmost setting for each marker corresponds to full network inference.

We employ 3 UCI (Dua & Graff, 2017) datasets of increasing size (input dimensionality, N. points): wine (11, 1439), kin8nm (8, 7373) and protein (9, 41157). We consider their standard train-test splits (Hernández-Lobato & Adams, 2015b) and their gap variants Foong et al. (2019b), designed to test for out of distribution uncertainty. For each split, we set aside 15% of the train set as a validation set. We use these for early stopping when finding the MAP estimates and for selecting the weights' prior precision. We keep other hyperparameters fixed across all models and datasets. Results are in Fig. 3.

We present in terms of log-likelihood (LL), as these take into account both accuracy and uncertainty. Larger models tend to perform better when doing MAP inference, with wine-gap and protein-gap being exceptions. We also find larger models improve over their respective MAP LLs more than small ones when performing approximate inference over the same number of weights. We conjecture

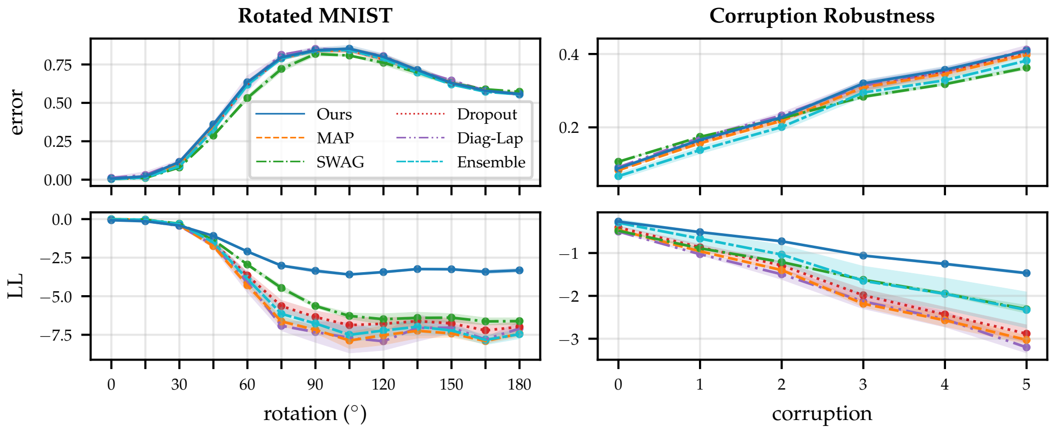

Figure 4: Results on the rotated MNIST (left) and the corrupted CIFAR (right) benchmarks of Ovadia et al. (2019), showing the mean $\pm$ std of the error (top) and log-likelihood (bottom) across three different seeds. Subnetwork inference retains better uncertainty calibration and robustness to distribution shift than point estimated networks and other Bayesian deep learning approaches.

this is due to an abundance of degenerate directions (weights) in the weight posterior of all models (Maddox et al., 2020). Full network inference in small models captures information about both useful and non-useful weights. In larger models, our pruning strategy allows us to dedicate a larger proportion of our resources to modelling informative weight variances and covariances. In 3 out of 6 datasets, we find abrupt increases in LL as we increase the number of weights over which we perform inference, followed by a plateau. Such plateaus might be explained by all of the most informative weight variances having already been accounted for. These results suggest **it is better to perform subnetwork inference in a large model than full network inference in a small one.**

5.3 Scaling to Image Classification with Distribution Shift

We now assess the robustness of large convolutional neural networks with subnetwork inference to distribution shift on image classification tasks compared to the following baselines: point-estimated networks (MAP), Bayesian deep learning methods that do less expressive inference over the full network: MC Dropout (Gal & Ghahramani, 2016), diagonal Laplace (both of which assume factorisation of the weight posterior), and SWAG (Maddox et al., 2019) (which assumes a diagonal plus low-rank posterior). We also benchmark deep ensembles (Lakshminarayanan et al., 2017). The latter is considered state-of-the-art for uncertainty quantification in deep learning (Ovadia et al., 2019; Ashukha et al., 2020). We use ensembles of 5 DNNs, as suggested by (Ovadia et al., 2019), and 16 samples for MC Dropout, diagonal Laplace and SWAG. We use a Dropout probability of 0.1 and a prior precision of $\lambda = 40,000$ for diagonal Laplace, found via grid search. We apply all approaches to ResNet-18 (He et al., 2016), which is composed of an input convolutional block, 8 residual blocks and a linear layer, for a total of $11,168,000$ weights. For subnetwork inference, we compute the linearized predictive distribution in Eq. (8) via the closed-form approximation for integrals between Gaussians and multi-class cross-entropy likelihoods described in (Gibbs, 1998). We use Wasserstein pruning to retain only 0.38% of the weights, yielding a subnetwork with only $42,438$ weights. This is the largest subnetwork for which we can tractably compute a full covariance matrix. Its size is $42,438^2 \times 4$Bytes ≈ 7.2GB. We use a prior precision of $\lambda = 500$, found via grid search. We perform the following two experiments, with results in Fig. 4.

Rotated MNIST: Following Ovadia et al. (2019); Antorán et al. (2020), we train all methods on MNIST and evaluate their predictive distributions on increasingly rotated digits. While all methods perform well on the original MNIST test set, their accuracy degrades quickly for rotations larger than 30 degrees. In terms of LL, ensembles perform best out of our baselines. Subnetwork inference obtains significantly larger LL values than all baselines, including ensembles. This suggests subnetwork inference is able to make accurate predictions in-distribution while assigning high uncertainty to out of stribution points. **Corrupted CIFAR:** Again following Ovadia et al. (2019); Antorán et al. (2020), we train models on CIFAR10 and evaluate them on data subject to 16 different corruptions with 5 levels of intensity each (Hendrycks & Dietterich, 2019). Subnetwork inference differentiates itself by being the least overconfident, outperforming on all baselines in terms of log-likelihood at all corruption levels. These results suggest **subnetwork inference results in better uncertainty calibration and robustness to distribution shift than other popular approaches**.

6 Related Work

Bayesian Deep Learning. There have significant efforts to characterise the posterior distribution over NN weights $p(\boldsymbol{W}|\mathcal{D})$. Hamiltonian Monte Carlo (Neal, 1995) remains the golden standard for approximate inference in BNNs to this day. Although asymptotically unbiased, sampling based approaches are difficult to scale to the large datasets (Betancourt, 2015). As a result, approaches which find the best surrogate posterior among an approximating family (most often Gaussians) have gained popularity. The first of these was the Laplace approximation, introduced by MacKay (1992), who also proposed approximating the predictive posterior with that of the linearised model (Khan et al., 2019; Immer et al., 2020). The popularisation of larger NN models has made surrogate distributions that capture correlations between weights computationally intractable. Thus, most modern methods make use of the mean field assumption (Blundell et al., 2015; Hernández-Lobato & Adams, 2015a; Gal & Ghahramani, 2016; Mishkin et al., 2018; Osawa et al., 2019). This comes at the cost of limited expressivity (Foong et al., 2019a) and empirical under-performance (Ovadia et al., 2019; Antorán et al., 2020) of uncertainty estimates. Our proposed approach recovers predictive posterior expressivity while maintaining tractability by lowering the dimensionality of the weight space considered. This allows us to scale up approximations that *do* consider weight correlations (MacKay, 1992; Louizos & Welling, 2016; Maddox et al., 2019; Ritter et al., 2018).

NN Linearisation. In the limit of infinite width, NNs converge to Gaussian Process (GP) behaviour (Neal, 1995; Matthews, 2017; Garriga-Alonso et al., 2018). Recently, these results have been extended to finite with BNNs when the surrogate posterior is Gaussian (Khan et al., 2019). We draw upon these results to formulate a subnetwork selection strategy for BNNs. Neural linear methods perform inference over only the last layer of a neural network, while keeping all other layers fixed (Riquelme et al., 2018; Ovadia et al., 2019; Snoek et al., 2015; Ober & Rasmussen, 2019; Pinsler et al., 2019; Kristiadi et al., 2020). These represent a different generalised linear model in which the basis functions are defined by the $l-1$ first layers of a NN. They can also be viewed as a special case of sub-network inference, in which only the last layer is kept.

Inference over Subspaces. The subfield of NN pruning aims to increase the computational efficiency of NNs by identifying the smallest subset of weights which are required to make accurate predictions. Approaches trade-off computational cost with compression efficiency ranging from those that require multiple training runs (Frankle & Carbin, 2019) to pruning before training (Wang et al., 2020). Our work differs in that it retains all NN weights but aims to find a small subset over which to perform probabilistic reasoning.More closely related work to ours is that of Izmailov et al. (2019), who propose to perform inference over a low-dimensional subspace of weights; e.g. one constructed from the principal components of the SGD trajectory. Moreover, several recent approaches use low-rank parameterizations of approximate posteriors in the context of variational inference (Rossi et al., 2019; Swiatkowski et al., 2020; Dusenberry et al., 2020). This could also be viewed as doing inference over an implicit subspace of weight space. In contrast, we propose a technique to find subsets of weights which are relevant to predictive uncertainty, i.e., axis aligned subspaces.

7 Conclusion

In this paper, we develop a *practical* and *scalable* method for expressive yet tractable probabilistic inference in deep neural networks. We approximate the posterior over a subset of the weights while keeping all other weights deterministic. Computational cost is decoupled from network size, allowing us to *scale* expressive approximations, such as full-covariance Gaussian distributions, to real-world sized NNs. The approach can be applied post-hoc to any pre-trained model, even when on a tight computational budget. Our empirical analysis suggests subnetwork inference 1) is more expressive and retains more uncertainty than comparably expensive approaches 2) allows us to employ larger NNs, which fit a broader range of functions, without sacrificing the quality of our uncertainty estimates 3) is competitive with state of the art uncertainty quantification methods, like deep ensembles (Lakshminarayanan et al., 2017), on real-world scale problems.

REFERENCES

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Javier Antorán, James Urquhart Allingham, and José Miguel Hernández-Lobato. Depth uncertainty in neural networks, 2020.

Arsenii Ashukha, Alexander Lyzhov, Dmitry Molchanov, and Dmitry Vetrov. Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. In *ICLR*, 2020.

Michael Betancourt. The fundamental incompatibility of scalable hamiltonian monte carlo and naive data subsampling. volume 37 of *Proceedings of Machine Learning Research*, pp. 533–540, Lille, France, 07–09 Jul 2015. PMLR. URL `http://proceedings.mlr.press/v37/betancourt15.html`.

Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Networks. In *Proceedings of The 32nd International Conference on Machine Learning (ICML)*, pp. 1613–1622, 2015.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*, 2017.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Michael W Dusenberry, Ghassen Jerfel, Yeming Wen, Yi-an Ma, Jasper Snoek, Katherine Heller, Balaji Lakshminarayanan, and Dustin Tran. Efficient and scalable bayesian neural nets with rank-1 factors. *arXiv preprint arXiv:2005.07186*, 2020.

Andrew YK Foong, David R Burt, Yingzhen Li, and Richard E Turner. On the expressiveness of approximate inference in bayesian neural networks. *arXiv*, pp. arXiv–1909, 2019a.

Andrew YK Foong, Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner. In-between uncertainty in bayesian neural networks. *ICML Workshop on Uncertainty and Robustness in Deep Learning*, 2019b.

Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

Yarin Gal. Uncertainty in deep learning. *University of Cambridge*, 1:3, 2016.

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

Adrià Garriga-Alonso, Carl Edward Rasmussen, and Laurence Aitchison. Deep convolutional networks as shallow gaussian processes. In *International Conference on Learning Representations*, 2018.

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

Mark N Gibbs. *Bayesian Gaussian processes for regression and classification*. PhD thesis, Citeseer, 1998.

Clark R Givens, Rae Michael Shortt, et al. A class of wasserstein metrics for probability distributions. *The Michigan Mathematical Journal*, 31(2):231–240, 1984.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1321–1330. JMLR. org, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

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

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic Backpropagation for Scalable Learning of Bayesian Neural Networks. In *ICML*, 2015b.

Alexander Immer, Maciej Korzepa, and Matthias Bauer. Improving predictions of bayesian neural networks via local linearization. *ICML Workshop on Uncertainty and Robustness in Deep Learning*, 2020.

Pavel Izmailov, Wesley J Maddox, Polina Kirichenko, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Subspace inference for bayesian deep learning. In *35th Conference on Uncertainty in Artificial Intelligence, UAI 2019*, 2019.

Mohammad Emtiyaz Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. *arXiv preprint arXiv:1806.04854*, 2018.

Mohammad Emtiyaz E Khan, Alexander Immer, Ehsan Abedi, and Maciej Korzepa. Approximate inference turns deep networks into gaussian processes. In *Advances in neural information processing systems*, pp. 3094–3104, 2019.

Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in neural information processing systems*, pp. 2575–2583, 2015.

Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being bayesian, even just a bit, fixes overconfidence in relu networks. *arXiv preprint arXiv:2002.10118*, 2020.

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

Neil David Lawrence. *Variational inference in probabilistic models*. PhD thesis, University of Cambridge, 2001.

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

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

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pp. 13132–13143, 2019.

Wesley J Maddox, Gregory Benton, and Andrew Gordon Wilson. Rethinking parameter counting in deep models: Effective dimensionality revisited. *arXiv preprint arXiv:2003.02139*, 2020.

James Martens. *Second-order optimization for neural networks*. University of Toronto (Canada), 2016.

James Martens and Ilya Sutskever. Learning recurrent neural networks with hessian-free optimization. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 1033–1040. Citeseer, 2011.

Alexander Graeme de Garis Matthews. *Scalable Gaussian process inference using variational methods*. PhD thesis, University of Cambridge, 2017.

Aaron Mishkin, Frederik Kunstner, Didrik Nielsen, Mark Schmidt, and Mohammad Emtiyaz Khan. Slang: Fast structured covariance approximations for bayesian deep learning with natural gradient. In *Advances in Neural Information Processing Systems*, pp. 6245–6255, 2018.

Radford M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, CAN, 1995. AAINN02676.

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*, pp. 427–436, 2015.

Sebastian W Ober and Carl Edward Rasmussen. Benchmarking the neural linear model for regression. *arXiv preprint arXiv:1912.08416*, 2019.

Kazuki Osawa, Siddharth Swaroop, Anirudh Jain, Runa Eschenhagen, Richard E Turner, Rio Yokota, and Mohammad Emtiyaz Khan. Practical deep learning with Bayesian principles. *arXiv preprint arXiv:1906.02506*, 2019.

Yaniv Ovadia, Emily Fertig, Balaji Lakshminarayanan, Sebastian Nowozin, D Sculley, Joshua Dillon, Jie Ren, Zachary Nado, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pp. 13969–13980, 2019.

Robert Pinsler, Jonathan Gordon, Eric Nalisnick, and José Miguel Hernández-Lobato. Bayesian batch active learning as sparse subset approximation. In *Advances in Neural Information Processing Systems*, pp. 6359–6370, 2019.

Carlos Riquelme, George Tucker, and Jasper Snoek. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. In *International Conference on Learning Representations*, 2018.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *International Conference on Learning Representations*, 2018.

Simone Rossi, Sebastien Marmin, and Maurizio Filippone. Walsh-hadamard variational inference for bayesian deep learning. *arXiv preprint arXiv:1905.11248*, 2019.

Nicol N Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. *Neural computation*, 14(7):1723–1738, 2002.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *International conference on machine learning*, pp. 2171–2180, 2015.

Jakub Swiatkowski, Kevin Roth, Bastiaan S Veeling, Linh Tran, Joshua V Dillon, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. The k-tied normal distribution: A compact parameterization of gaussian mean field posteriors in bayesian neural networks. *arXiv preprint arXiv:2002.02655*, 2020.

Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SkgsACVKPH`.