3rd Symposium on Advances in Approximate Bayesian Inference, 2020 1–19

Bayesian Evidential Deep Learning with PAC Regularization

Manuel Haußmann MANUEL.HAUSSMANN@IWR.UNI-HEIDELBERG.DE
HCI/IWR, Heidelberg University, Germany

Sebastian Gerwinn SEBASTIAN.GERWINN@DE.BOSCH.COM
Melih Kandemir MELIH.KANDEMIR@DE.BOSCH.COM
Bosch Center for Artificial Intelligence, Renningen, Germany

Abstract

We propose a novel method for closed-form predictive distribution modeling with neural nets. In quantifying prediction uncertainty, we build on Evidential Deep Learning, which has been impactful as being both simple to implement and giving closed-form access to predictive uncertainty. We employ it to model aleatoric uncertainty and extend it to account also for epistemic uncertainty by converting it to a Bayesian Neural Net. While extending its uncertainty quantification capabilities, we maintain its analytically accessible predictive distribution model by performing progressive moment matching for the first time for approximate weight marginalization. The eventual model introduces a prohibitively large number of hyperparameters for stable training. We overcome this drawback by deriving a vacuous PAC bound that comprises the marginal likelihood of the predictor and a complexity penalty. We observe on regression, classification, and out-of-domain detection benchmarks that our method improves model fit and uncertainty quantification.

1. Introduction

As the interest of the machine learning community in data-efficient and uncertainty-aware predictors increases, research on Bayesian Neural Networks (BNNs) (MacKay, 1995; Neal, 1995) gains prominence. Differently from deterministic nets, BNNs have stochastic weights. Thanks to their stacked structure, they propagate predictive uncertainty through the hidden layers and can characterize complex uncertainty structures. Exact inference of such a highly nonlinear system is analytically intractable and very hard to approximate with high precision. Consequently, most research on BNNs thus far focused on improving approximate inference techniques in terms of precision and computational cost (Hernández-Lobato and Adams, 2015; Kingma et al., 2015; Louizos and Welling, 2017). These approaches take the posterior inference of global parameters as given and develop their approximation based on it. A newly emerging alternative approach is direct predictive distribution modeling. It proposes devising a highly expressive predictive distribution with several free parameters. These parameters are then fit to data via maximum likelihood estimation. This way, observations are used to train directly the end product of interest: the predictive distribution, bypassing the need for an intractable posterior inference step. Some existing methods model the predictive distribution via a stochastic process parameterized as a neural net (Garnelo et al., 2018a), while others introduce local priors on the likelihood activations, integrate them out and train the hyperparameters of the marginal (Sensoy et al., 2018; Malinin and Gales, 2018).

This work builds on Evidential Deep Learning (EDL) (Sensoy et al., 2018), due to its technical simplicity and observed effectiveness. EDL places a Dirichlet prior on the class assignment probabilities of a classifier and parameterizes the Dirichlet strengths of this prior with a neural net. While demonstrating substantial improvements in out-of-domain detection and adversarial robustness, it is not capable of decomposing epistemic and aleatoric uncertainties. We propose an efficient and effective method that extends EDL to BNNs, equipping it with more advanced uncertainty quantification and decomposition capabilities. We assume independent local weight random variables controlling the BNN for each input/target pair of data points, which share common hyperparameters. Marginalizing these data-point specific weights of our network we perform training via type 2 maximum likelihood (ML) (Berger, 1985) on the prior hyperparameters. This analytically intractable marginalization is approximated using the Central Limit Theorem (CLT). Differently from other weight marginalization approaches that assign global weight distributions on infinitely many neurons and recover Gaussian Processes (GP) (Neal, 1995; Lee et al., 2018; Garragia-Alonso et al., 2019), our formulation maintains finitely many hidden units per layer and assigns them individual weight distributions. Due to per data-point treatment of the weight marginalization the BNN scales linearly with the data size. The advantages of such a BNN with data-point specific marginalization comes at the expense of a major drawback. The number of hyperparameters in a weight-marginalized BNN grows proportionally to the number of synaptic connections. Maximizing the marginal likelihood w.r.t. such a large number of hyperparameters is prone to overfitting (Bauer et al., 2016). Since the weight variables are marginalized out and their hyperparameters of the weight prior are set via optimization, the model can no longer incorporate regularizing knowledge other than the parametric form of the prior distribution (e.g. normal with mean and variance as free parameters). We address this drawback by deriving a provably vacuous Probably Approximately Correct (PAC) (McAllester, 1999, 2003) bound that contains the marginal likelihood as its empirical risk term. Minimization of this PAC bound automatically balances the fit to the data and deviation from a prior regularizing hypothesis.

We compare our method on various standard regression, classification, and out-of-domain detection benchmarks against state-of-the-art approximate posterior inference based BNN training approaches. We observe that our method provides competitive prediction accuracy and better uncertainty estimation scores than those baselines.

2. Bayesian Evidential Deep Learning

Evidential Deep Learning. Classification with cross-entropy loss in deep learning can be interpreted as a categorical likelihood parameterized by a neural net. EDL generalizes this setup by parameterizing a prior by a neural net $f(\cdot;\mathbf{w})$ with deterministic weights $\mathbf{w}$, instead of the likelihood. For classification, given a data set $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)_{n=1}^N\}$ consisting of N pairs of input $\mathbf{x}_n$ and target $\mathbf{y}_n$, a natural choice for the prior $p(\boldsymbol{\lambda}_n|\mathbf{w}, \mathbf{x}_n)$ on the categorical likelihood is a Dirichlet distribution, and the final model becomes

$$\boldsymbol{\lambda}_n|\mathbf{x}_n \sim \mathcal{D}ir\big(\boldsymbol{\lambda}_n|\boldsymbol{\alpha}_n\big), \qquad \mathbf{y}_n|\boldsymbol{\lambda}_n \sim \mathcal{C}at\big(\mathbf{y}_n|\boldsymbol{\lambda}_n\big), \qquad \forall n, \tag{1}$$

where $\boldsymbol{\alpha}_n = f(\mathbf{x}_n; \mathbf{w}) + 1$. This way, the model explicitly accounts for the *distributional uncertainty* which may arise due to a mismatch between the train and test data distributions. To train, the loss is the expected sum of squares between $\mathbf{y}_n$ and $\boldsymbol{\lambda}_n$ with an additional regularizing Kullback Leibler divergence on the $\boldsymbol{\lambda}_n$.

Bayesian Local Neural Nets. Parameterizing the likelihood by a BNN $f(\cdot;\mathbf{w})$ with random variables $\mathbf{w}$ as the weights results in the following probabilistic model

$$\mathbf{w}_n \sim p_\phi(\mathbf{w}_n), \qquad \mathbf{y}_n|\mathbf{x}_n,\mathbf{w}_n \sim p(\mathbf{y}_n|\mathbf{x}_n,\mathbf{w}_n), \qquad \forall n$$

where $p(\mathbf{y}_n|\mathbf{x}_n,\mathbf{w}_n)$ is some likelihood, and $p_\phi(\mathbf{w}_n)$ is a prior over local weights with shared hyperparameters ϕ. Differently from a canonical BNN where all data points share the same global weight latent variable (Blundell et al., 2015; Gal and Ghahramani, 2015; Kingma et al., 2015; Louizos and Welling, 2017), here the mapping between each input-output pair is determined by a separate random variable $\mathbf{w}_n$, giving a unique mapping constrained by sharing a common set of prior hyperparameters ϕ, which is required for the *type 2 ML* based objective we introduce below. As this modification implies a collection of only local latent variables, we refer to the resulting model as a *Bayesian Local Neural Net (BLNN)*. It consists of two sources of uncertainty. First, the *model (epistemic) uncertainty* captured by the prior over the parameters, i.e. $p_\phi(\mathbf{w}_n)$, which accounts for the mismatch between the model and the true functional mapping from $\mathbf{x}_n$ to $\mathbf{y}_n$. Second, the irreducible *data (aleatoric) uncertainty* given by $p(\mathbf{y}_n|\mathbf{x}_n,\mathbf{w}_n)$ stemming from irreducible measurement noise.

BLNN Training with Type 2 ML and Prediction. An alternative to full Bayesian model training with posterior inference is marginalizing out all latent variables and maximizing the marginal likelihood with respect to the hyperparameters, hence avoiding the posterior inference step on latent variables, referred to as *type 2 ML* (Berger, 1985). The optimization objective then is: $\arg\max_\phi \log p_\phi(\mathbf{y}|\mathbf{X})$. Introducing an independent $\mathbf{w}_n$ for each pair $(\mathbf{x}_n,\mathbf{y}_n)$ leads to a sum of N independent marginal likelihoods,

$$\log p_\phi(\mathbf{y}|\mathbf{X}) = \sum_{n=1}^{N} \log \int p(\mathbf{y}_n|\mathbf{x}_n,\mathbf{w}_n)p_\phi(\mathbf{w}_n)d\mathbf{w}_n = \sum_{n=1}^{N} \log p_\phi(\mathbf{y}_n|\mathbf{x}_n) \tag{2}$$

which is amenable to using mini-batches for further scalability. The marginal of a training data point is identical to the posterior predictive for new test data $\mathbf{x}^*$. Hence, an analytic approximation developed for training is directly applicable to test time.

Analytic Marginalization of Local Weights with Moment Matching. Marginalizing out the local weights $\mathbf{w}_n$ in (2) is an intractable problem due to the highly nonlinear neural net appearing in the likelihood $p(\mathbf{y}_n|\mathbf{x}_n,\mathbf{w}_n)$. However, we can marginalize the weights approximately by recursive moment matching resorting to the Central Limit Theorem (CLT). This technique has previously been used in BNNs for other purposes, such as expectation propagation (Hernández-Lobato and Adams, 2015; Ghosh et al., 2016), fast dropout (Wang and Manning, 2013), and variational inference (Wu et al., 2019). We employ the same technique for marginalizing out the weights of the BLNN (see appendix).

Bayesian Evidential Deep Learning. The original formulation for EDL builds on deterministic nets, hence assumes a point estimate on the weights (consciously ignoring model uncertainty) and a sum-of-squares loss term. We improve this framework by assigning a local prior on the EDL weights $p_\phi(\mathbf{w}_n)$,[1] and instead consider the optimization of the marginal likelihood, which amounts to employing a BLNN as a prior on the likelihood and

1. Throughout this work we assume $\mathbf{w}_n \sim p_\phi(\mathbf{w}_n) = \mathcal{N}(\mathbf{w}_n|\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$, i.e. $\phi = (\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$.

marginalizing over all $\mathbf{w}_n$ as well as $\boldsymbol{\lambda}_n$. We name the eventual model that combines BLNN with EDL *Bayesian Evidential Deep Learning (BEDL)*. By virtue of the localized weights, the marginal likelihood of BEDL factorizes across data points, bringing additive data point specific marginal log-likelihoods maintaining the central source of its scalability, formally

$$\log p_\phi(\mathbf{y}|\mathbf{X}) = \sum_n \log \int p(\mathbf{y}_n|\boldsymbol{\lambda}_n)p(\boldsymbol{\lambda}_n|\mathbf{w}_n, \mathbf{x}_n)p_\phi(\mathbf{w}_n)d\boldsymbol{\lambda}_n d\mathbf{w}_n = \sum_n \log p_\phi(\mathbf{y}_n|\mathbf{x}_n). \quad (3)$$

The marginalization of $\boldsymbol{\lambda}_n$ on the last step can be performed analytically under conjugacy, efficiently approximated by Taylor expansion, or via Monte Carlo sampling, after moment matching to marginalize the weights $\mathbf{w}_n$. For the C-class classification task with one-hot encoded targets $\mathbf{y}_n$ and Dirichlet distributed $\boldsymbol{\lambda}_n$, this gives us for the n-th term in (3),

$$\log \int \int p(\mathbf{y}_n|\boldsymbol{\lambda}_n)p(\boldsymbol{\lambda}_n|\boldsymbol{\alpha}_n)d\boldsymbol{\lambda}_n \mathcal{N}(\mathbf{f}_n^L|\mathbf{m}_n, \mathbf{s}_n^2)d\mathbf{f}_n^L = \log \mathbb{E}_{\mathcal{N}(\mathbf{f}_n^L|\mathbf{m}_n,\mathbf{s}_n^2)}\left[\prod_{c=1}^C \left(\frac{\alpha_{nc}}{\alpha_{n0}}\right)^{y_{nc}}\right], \quad (4)$$

where $\mathbf{f}_n^L$ is the last layer after marginalization of the others (see Appendix D), and we use $\boldsymbol{\alpha}_n = (\alpha_{n1}, ..., \alpha_{nC}) = \exp(\mathbf{f}_n^L)$, and $\alpha_{n0} = \sum_c \alpha_{nc}$. Since the computational bottleneck on weight marginalizing is circumvented by the analytical CLT-based moment matching, the final expectation can be efficiently approximated by samples.

3. A Vacuous PAC Bound to Regularize BEDL

Training the objective in (3) is effective for fitting a predictor on data. It also naturally provides a learned loss attenuation mechanism. However, it lacks a key advantage of the Bayesian modeling paradigm. As the hyperparameters of the weight priors are employed for model fitting, they no longer contribute to training as complexity penalizers. It is well-known from the GP literature that marginal likelihood-based training is prone to overfitting for models with a large number of hyperparameters (Bauer et al., 2016).[2] We address this shortcoming by complementing the marginal likelihood objective of (3) with a penalty term derived from learning-theoretic first principles. We tailor the eventual loss only for robust model training and keep it maximally generic across learning setups. This comes at the expense of arriving at a generalization bound that makes a theoretically trivial statement, yet brings significant improvements to training quality as illustrated in our experiments.

PAC bounds have been commonly used for likelihood-free and loss-driven learning settings. A rare exception by Germain et al. (2016) proves the theoretical equivalence of a particular sort of PAC bound to variational inference. Similarly, we keep the notion of a likelihood in our risk definition, but differently, we correspond our bound to the marginal likelihood. Given a predictor h chosen from a hypothesis class $\mathcal{H}$ as a mapping from $\mathbf{x}$ to $\mathbf{y}$, we define the true and the empirical risks as

$$R(h) = -\mathbb{E}_{\mathbf{x},\mathbf{y}\sim\Delta}\left[p\big(\mathbf{y}|h(\mathbf{x})\big)\right] \qquad \text{and} \qquad R_{\mathcal{D}}(h) = -\frac{1}{N}\sum_{n=1}^N p\big(\mathbf{y}_n|h(\mathbf{x}_n)\big),$$

2. A direct objection is that one could go one level higher in the hierarchy, introducing hyperpriors over the parameters ϕ. We derive in the appendix how one could do this, but preliminary experiments have shown it to perform a lot worse than the PAC-based approach.

for the data set $\mathcal{D}$ drawn from an arbitrary and unknown data distribution Δ. The risks $R(h)$ and $R_{\mathcal{D}}(h)$ are bounded below by $-\max p(y|h(x))$ and above by zero. Although this setting relaxes the common assumption that bounds risk to the $[0, 1]$ interval, it is substantially simpler than the one suggested in (Germain et al., 2016), which defines $R(h) = \mathbb{E}_{x,y\sim\Delta}\left[\log p(y|h(x))\right] \in (-\infty, +\infty)$. This unboundedness brings severe technical complications, which are no longer relevant for our approach. Denoting by Q the distributions learnable over $\mathcal{H}$ and by P some regularizing distribution on $\mathcal{H}$, according to Theorem 2.1[3] in (Germain et al., 2009) we have for any $\delta \in (0, 1]$ and any convex function $d(\cdot, \cdot)$

$$\Pr\left\{ d\Big(\mathbb{E}_{h\sim Q}\left[R_{\mathcal{D}}(h)\right], \mathbb{E}_{h\sim Q}\left[R(h)\right]\Big) \leq \frac{\mathrm{KL}\left(Q \parallel P\right) + \log(B/\delta)}{N} \right\} \geq 1-\delta, \tag{5}$$

where $B = \mathbb{E}_{\mathcal{D}\sim\Delta}\left[\mathbb{E}_{h\sim P}\left[\exp\left(Nd(R_{\mathcal{D}}(h), R(h))\right)\right]\right]$. Using a quadratic distance measure $d(x, y) = (x - y)^2$ and suitably bounding B by exploiting the boundedness of the likelihood and in turn the risk, we get as an upper bound on the expected true risk (see the appendix)

$$-\frac{1}{N}\sum_{n=1}^{N}\log p_{\phi}(\mathbf{y}_n|\mathbf{x}_n) + \sqrt{\frac{\mathrm{KL}\left(Q \parallel P\right) - \log\delta}{N} + \frac{\log\max(B)}{N}}, \tag{6}$$

which is the objective we use to train BEDL that contains the marginal likelihood in the first term and a regularizer in the second. The additional term resembles the KL term in the EDL loss, and gives a theoretically-grounded mechanism to incorporate regularization.

Computation of the Bound for Classification. For a C-class classification we can compute the first term in (6) as discussed above ((4)). In the regularization term as we use (similar to EDL) $P = \mathcal{D}ir\big(\boldsymbol{\lambda}|(1, \ldots, 1)\big)$, i.e. the assumption that each class is equally likely, as the regularizing distribution. Given $Q = \mathcal{D}ir(\boldsymbol{\lambda}|\boldsymbol{\alpha})$, $\mathrm{KL}\left(Q \parallel P\right)$ is analytically tractable, as is the last term where we use the upper bound $\log\max(B) \leq N$.

Looseness of the Bound. It should be noted that while we use PAC theory to derive and motivate the final objective, it should no longer be used in its PAC interpretation, as the approximations result in a loose bound that is trivially fulfilled. Its justification lies mainly in its regularizing function.

4. Experiments

We evaluate *BEDL* and its PAC-regularized version *BEDL+Reg* on several classification tasks. Complete details on the training procedure can be found in the appendix. Additionally we provide there a discussion of the computational cost as well as extensions to regression and a comparison to GP based approaches.

Classification and out-of-domain detection. We train LeNet-5 networks on the MNIST train split, evaluate their classification accuracy on the MNIST test split as the in-domain task, and measure their uncertainty on the Fashion-MNIST data set as the out-of-domain task, adhering to the protocol used in prior work (Louizos and Welling, 2017; Sensoy et al., 2018).[4]

3. The theorem assumes risks to be defined within the $[0, 1]$ interval in the original paper, but it is valid for any bounded risk. Our risk definitions can trivially be squashed into $[0, 1]$ up to a constant.
4. Due to the license status of the not-MNIST data conflicting with the affiliation of the authors, we have to change the setup of earlier work, using instead Fashion-MNIST as the closest substitute.

		MNIST (In Domain)	Fashion-MNIST (Out-of-Domain)	CIFAR 1-5 (In Domain)	CIFAR 6-10 (Out-of-Domain)
	Reference	Test Error (%)	ECDF-AUC	Test Error(%)	ECDF-AUC
MC Dropout	(Gal and Ghahramani, 2016)	1.12	0.429	**18.36**	0.946
VarOut	(Kingma et al., 2015)	1.47	1.381	33.94	0.673
DVI	(Wu et al., 2019)	0.72	1.318	23.32	1.251
EDL	(Sensoy et al., 2018)	1.08	0.132	20.34	0.451
BEDL	Ours	0.81	1.512	24.38	1.253
BEDL+Reg	Ours	**0.66**	**0.055**	20.02	**0.083**

Table 1: **Classification and OOD Detection.** Test error and area under the curve of the empirical CDF (ECDF-AUC) of the predictive entropies on two pairs of datasets.

We expect from a perfect model to predict true classes with high accuracy on the in-domain task and always predict a uniform probability mass on the out-of-domain task, i.e. the area under the curve of the empirical CDF (ECDF-AUC) of its predictive distribution entropy is zero. We perform the same experiment on CIFAR10 using the first five classes for the in-domain task and treating the rest as out-of-domain. We use $P := \mathcal{D}ir\big(\boldsymbol{\lambda}|(1,\ldots,1)\big)$ as the regularization prior on the class assignment parameters, which has the uniform probability mass on its mean, encouraging an OOD signal in the absence of contrary evidence. In Table 1, we compare BEDL+Reg against EDL (Sensoy et al., 2018), also the non-Bayesian and heuristically trained counterpart of BEDL+Reg. We consider EDL also as a special case of Prior Networks (Malinin and Gales, 2018) that does not need to rely on OOD data during training time, commensurate for our training assumptions. We evaluate MC Dropout, VarOut, and DVI as baselines in this setup. BEDL+Reg improves the state of the art in all four metrics except the CIFAR10 in-domain task, where it ranks second after the prediction time weight sampling-based (hence less scalable) MC Dropout. Remarkably, BEDL+Reg detects the OOD samples with significantly better calibrated ECDF-AUC scores than EDL.

5. Conclusion

We present a method for performing Bayesian inference within the framework of evidential deep learning. Employing type 2 maximum likelihood for inference and combining it with PAC-bounds for regularization, we achieve higher accuracy and better predictive uncertainty estimates while maintaining scalable inference. Exact inference in a fully Bayesian model such as a GP (c.f. Table 2) or Hamiltonian Monte Carlo inference for BNNs (Bui et al., 2016) are known to provide better error rates and test log-likelihood scores, yet their computational demand does not scale well to large networks and data-sets. Our method, on the other hand, shows strong indicators for improvement in uncertainty quantification and predictive performance when compared to other BNN approximate inference schemes with reasonable computational requirements. These benefits of the BEDL+Reg framework might especially be fruitful in setups such as model-based deep reinforcement learning, active learning, and data synthesis, where uncertainty quantification is a vital ingredient of the predictor.

References

M. Bauer, M. v.d. Wilk, and C.E. Rasmussen. Understanding Probabilistic Sparse Gaussian Process Approximations. In *NIPS*, 2016.

James O Berger. *Statistical decision theory and Bayesian analysis.* Springer Science & Business Media, 1985.

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wiestra. Weight Uncertainty in Neural Networks. In *ICML*, 2015.

T. Bui, D. Hernández-Lobato, J. M. Hernandez-Lobato, Y. Li, and R. Turner. Deep Gaussian Processes for Regression using Approximate Expectation Propagation. In *ICML*, 2016.

O. Catoni. PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning. *IMS Lecture Notes Monograph Series*, 56, 2007.

D. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

S. Depeweg, J.-Miguel Hernandez-Lobato, F. Doshi-Velez, and S. Udluft. Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning. In *ICML*, 2018.

G. Dziugaite and D.M. Roy. Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data. In *UAI*, 2017.

B.J. Frey and G.E. Hinton. Variational Learning in Nonlinear Gaussian Belief Networks. *Neural Computation*, 11(1):193–213, 1999.

Y. Gal and Y. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *ICML*, 2016.

Y. Gal and Z. Ghahramani. Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference. *arXiv preprint arXiv:1506.02158*, 2015.

M. Garnelo, D. Rosenbaum, C. J Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. J Rezende, and SM Eslami. Conditional neural processes. *arXiv preprint arXiv:1807.01613*, 2018a.

M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J Rezende, SM Eslami, and Y. W. Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018b.

A. Garragia-Alonso, C.E. Rasmussen, and L. Aitchinson. Deep Convolutional Networks as Shallow Gaussian Processe. In *ICLR*, 2019.

P. Germain, A. Lacasse, F. Laviolette, and M. Marchand. PAC-Bayesian Learning of Linear Classifiers. In *ICML*, 2009.

P. Germain, F. Bach, A. Lacoste, and S. Lacoste-Julien. PAC-Bayesian Theory Meets Bayesian Inference. In *NIPS*, 2016.

S. Ghosh, J. Yedidia, and F.M. Delle Fave. Assumed Density Filtering Methods for Scalable Learning of Bayesian Neural Networks. In *AAAI*, 2016.

J. M. Hernández-Lobato and R. Adams. Probabilistic Backpropagation for Scalable Learning of Bayesian Neural Networks. In *ICML*, 2015.

D.P. Kingma, T. Salimans, and M. Welling. Variational Dropout and The Local Reparameterization Trick. In *NIPS*, 2015.

B. Lakshminarayanan, A. Pritzel, and C/ Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *NIPS*, 2017.

J. Lee, Y. Bahri, R. Novak, S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep Neural Networks as Gaussian Processe. In *ICLR*, 2018.

C. Louizos and M. Welling. Multiplicative Normalizing Flows for Variational Bayesian Neural Networks. In *ICML*, 2017.

D.J. MacKay. Probable Networks and Plausible Predictions – A Review of Practical Bayesian Methods for Supervised Neural Networks. *Network: Computation in Neural Systems*, 6(3): 469–505, 1995.

A. Malinin and M. Gales. Predictive uncertainty estimation via prior networks. In *NeurIPS*, 2018.

A. G. de G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani. Gaussian process behaviour in wide deep neural networks. In *ICLR*, 2018.

D. McAllester. PAC-Bayesian Model Averaging. In *COLT*, 1999.

D. McAllester. PAC-Bayesian Stochastic Model Selection. *Machine Learning*, 51:5–21, 2003.

D. Molchanov, A. Ashukha, and D. Vetrov. Variational Dropout Sparsifies Deep Neural Networks. In *ICML*, 2017.

R. Neal. *Bayesian Learning for Neural Networks.* PhD Thesis, 1995.

M. Seeger. PAC-Bayesian Generalisation Error Bounds for Gaussian Process Classification. *Journal of Machine Learning Research*, 3:233–269, 2002.

M. Sensoy, L. Kaplan, and M. Kandemir. Evidential Deep Learning to Quantify Classification Uncertainty. In *NeurIPS*, 2018.

J.T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for Simplicity: The All Convolutional Net. In *ICLR*, 2015.

S.I. Wang and C.D. Manning. Fast Dropout Training. In *ICML*, 2013.

A. Wu, S. Nowozin, E. Meeds, R.E. Turner, J.M. Hernández-Lobato, and A.L. Gaunt. Deterministic Variational Inference for Bayesian Neural Networks. In *ICLR*, 2019.

Appendix A. Related Work

CLT-based moment matching. The objective for variational inference on BNNs (Kingma et al., 2015; Wu et al., 2019; Gal and Ghahramani, 2016), optimizing a global variational posterior $q(\mathbf{w})$, consists of a computationally intractable $\mathbb{E}_{q(\mathbf{w})}\left[\log p(\mathbf{y}|\mathbf{x}, \mathbf{w})\right]$ that decomposes across data points. Fast dropout (Wang and Manning, 2013) approximates these terms via local reparameterization with moment matching. The same local reparameterization has been later combined with a KL term to perform mean-field VI via MC sampling (Kingma et al., 2015) or moment matching (Wu et al., 2019). Our BLNN formulation is akin to an amortized VI approach learning a single global set of posterior parameters ϕ for the variational posterior approximation $q_\phi(\mathbf{w})$. We use the same trick to marginalize the local weights, which keeps the machinery intact until the top-most step where the order of the $\log(\cdot)$ and $\mathbb{E}\left[\cdot\right]$ operations is swapped. This small change, however, has a large impact on the quality of uncertainty estimations.

Wide neural nets as GPs. The equivalence of a GP to a weight-marginalized BNN with a single infinitely wide hidden layer has been discovered long ago (Neal, 1995). This result has later been generalized to multiple dense layers (Lee et al., 2018; Matthews et al., 2018), as well as to convolutional layers (Garragia-Alonso et al., 2019). The asymptotic treatment of the neuron count makes this approach exact at the expense of a lack of neuron-specific parameterization. The eventual GP has few hyperparameters to train, however, a prohibitively expensive covariance matrix to calculate. We employ the same training method on a middle ground where the hyperparameter count is double as many as a deterministic net and the cross-covariances across data points are not explicitly modeled.

Predictive density modeling. Neural Processes (Garnelo et al., 2018a,b) follow a GP inspired approach of learning the predictive density using neural networks as part of the mapping from input to output space relying on an incorporation of the input data as context points for the predictive distribution of a test point. Earlier work on prior networks (Malinin and Gales, 2018) parameterizes a prior to a classification-specific likelihood with deterministic neural nets, hence, discards model uncertainty. Additionally, they require samples from another domain to learn the distributional awareness. BEDL reformulates prior networks independently from the output structure, extends them to support also model uncertainty, and introduces a principled scheme for their training.

Appendix B. Uncertainty Decomposition.

Following Depeweg et al. (2018) one can use the law of total variance to decompose the predictive variance for one data point $(\mathbf{x}, \mathbf{y})$ as follows

$$\mathrm{var}\left[\mathbf{y}|\mathbf{x}\right] = \mathrm{var}_{\mathbf{w}}\left[\mathbb{E}\left[\mathbf{y}|\mathbf{w}, \mathbf{x}\right]\right] + \mathbb{E}_{\mathbf{w}}\left[\mathrm{var}\left[\mathbf{y}|\mathbf{w}, \mathbf{x}\right]\right],$$

where the first term, $\mathrm{var}_{\mathbf{w}}\left[\mathbb{E}\left[\mathbf{y}|\mathbf{w}, \mathbf{x}\right]\right]$, focuses on the contribution to this predictive uncertainty by the variance over the network weights, i.e. the epistemic uncertainty, while the second, $\mathbb{E}_{\mathbf{w}}\left[\mathrm{var}\left[\mathbf{y}|\mathbf{w}, \mathbf{x}\right]\right]$, represents the remaining variance in the likelihood for the average weights. After having marginalized over $\boldsymbol{\lambda}$, the mean and variance $\mathbb{E}\left[\mathbf{y}|\mathbf{w}, \mathbf{x}\right]$, $\mathrm{var}\left[\mathbf{y}|\mathbf{w}, \mathbf{x}\right]$, are analytically tractable with

$$\mathbb{E}\left[y_c|\mathbf{w},\mathbf{x}\right] = \frac{\alpha_c}{\alpha_0} \quad \text{and} \quad \operatorname{var}\left[y_c|\mathbf{w},\mathbf{x}\right] = \frac{\alpha_c}{\alpha_0}\Big(1 - \frac{\alpha_c}{\alpha_0}\Big).$$

The EDL formulation allows for an analytic computation of the predictive variance, however as it considers only deterministic weights, it gets for learned parameters $\hat{\mathbf{w}}$

$$\operatorname{var}\left[\mathbf{y}|\mathbf{x}\right] = \operatorname{var}\left[\mathbf{y}|\hat{\mathbf{w}},\mathbf{x}\right],$$

i.e. only a measure of the aleatoric uncertainty, lacking the epistemic. Our extension allows for the decomposition of the predictive uncertainty maintaining analytical tractability of the approximation to a great extent

$$\operatorname{var}\left[\mathbf{y}|\mathbf{x}\right] = \operatorname{var}_{\mathbf{w}}\left[\mathbb{E}\left[\mathbf{y}|\mathbf{w},\mathbf{x}\right]\right] + \mathbb{E}_{\mathbf{w}}\left[\operatorname{var}\left[\mathbf{y}|\mathbf{w},\mathbf{x}\right]\right] \approx \operatorname{var}_{\mathbf{f}^L}\left[\mathbb{E}\left[\mathbf{y}\middle|\mathbf{f}^L,\mathbf{x}\right]\right] + \mathbb{E}_{\mathbf{f}^L}\left[\operatorname{var}\left[\mathbf{y}\middle|\mathbf{f}^L,\mathbf{x}\right]\right],$$

where the final variance and expectation can be efficiently approximated with samples as discussed above.

Appendix C. Extensions

C.1. BEDL with Hyperpriors

Instead of relying on the PAC-bound based could try to incorporate further hyper-priors on ϕ. This would hopefully reintroduce the missing regularization BEDL faces. The hierarchical model then has the following structure:

$$\begin{aligned}
\phi &\sim p(\phi),\\
\mathbf{w}|\phi &\sim \prod_n p_\phi(\mathbf{w}_n),\\
\boldsymbol{\lambda}|\mathbf{w},\mathbf{x} &\sim \prod_n p(\boldsymbol{\lambda}_n|\mathbf{w}_n,\mathbf{x}_n),\\
\mathbf{y}|\boldsymbol{\lambda} &\sim \prod_n p(\mathbf{y}_n|\boldsymbol{\lambda}_n).
\end{aligned}$$

The marginal to be optimized over is then given as

$$\log p(\mathbf{y},\phi|\mathbf{X}) = \sum_n \log \int p(\mathbf{y}_n|\boldsymbol{\lambda}_n)p(\boldsymbol{\lambda}_n|\mathbf{w}_n,\mathbf{x}_n)p(\mathbf{w}_n|\phi)d\lambda_n d\mathbf{w}_n + \log p(\phi).$$

The first term is our regular marginal likelihood, while the second serves as as a regularizer as an optimization scheme aims to choose ϕ such that the marginal likelihood is high, but also that the prior density is large. The form of this hyperprior will vary depending on the problem at hand, but if we consider e.g. the i-th weight of the BNN w_n^i to follow a normal distribution, we have

$$w_n^i|\phi_i \sim \mathcal{N}(w_n^i|\mu_i,\sigma_i^2), \qquad \text{where } \phi_i = (\mu_i,\sigma_i^2).$$

An obvious choice for the prior $p(\phi_i)$ is then given as

$$p(\phi_i) = p(\mu_i)p(\sigma_i^2) = \mathcal{N}(\mu|0,\alpha_0^{-1})\mathcal{I}nv\mathcal{G}am(\sigma^2|a_0,b_0).$$

The regression results summarized in Table 1 in the main paper however show that this approach tends to perform worse than both the PAC regularized BEDL as well as the unregularized BEDL.

C.2. Generalization to Regression

The EDL formulation was introduced by Sensoy et al. (2018) only for the case of classifications. However, the main motivation of the approach can also be extended to the case of regression. We place a normal likelihood over the targets, treating the λ_n as the mean and another normal as the distribution over λ_n parameterizing both mean and variance with a BNN, giving

$$\begin{aligned}\mathbf{w}_n &\sim p_\phi(\mathbf{w}_n)\\ \lambda_n|\mathbf{w}_n, \mathbf{x}_n &\sim \mathcal{N}\Big(\lambda_n\Big|f_1(\mathbf{x}_n;\mathbf{w}_n), \exp\left(f_2(\mathbf{x}_n;\mathbf{w}_n)\right)\Big)\\ y_n|\lambda_n &\sim \mathcal{N}\left(y_n|\lambda_n, \beta^{-1}\right),\end{aligned}$$

with some fixed observation precision β. For the n-th sample in (3) with $\mathbf{f}_n^L = (f_{n1}^L, f_{n2}^L)$, $\mathbf{m}_n = (m_{n1}, m_{n2})$, and $\mathbf{s}_n^2 = (s_{n1}^2, s_{n2}^2)$ (the moment matching mean and variance), the log marginal likelihood is then given as

$$\log p(y_n|\mathbf{x}_n) = \log \mathcal{N}\Big(y_n\Big|m_{n1}, \beta^{-1} + s_{n1}^2 + \exp(m_{n2} + s_{n2}^2/2)\Big).$$

The approximation is computed via a final moment matching step approximating the result of the inner integral $p(\lambda_n) = \int p(\lambda_n|\mathbf{f}_n^L)p(\mathbf{f}_n^L)d\mathbf{f}_n^L$ with a normal distribution while the ultimate equality follows directly from standard results on normal distributions. Contrary to the case of classification where the final step requires samples, the regression stays sampling-free. We bound $\max B$ in (6) by exploiting that β is fixed prior to training. Consequently, we get $\frac{\log \max(B)}{N} \leq \frac{\beta}{2\pi}$ as a bound, which, as for the classification case, gives only a trivial performance guarantee (exceeding the maximum possible risk) but provides a justified training scheme.

C.3. Further Extensions

Adaptations of CLT-based recursive moment matching to many other activation types and skip connections are feasible without further approximations. Max pooling can also be incorporated using approximations, but have also been shown to be replaceable altogether by strided convolutions without a performance loss (Springenberg et al., 2015). Deeper networks tend to require normalization procedures, which are not directly amenable to this moment matching. However tractable moments can also be computed for activation functions such as the ELU (Clevert et al., 2015) as we show in the appendix alleviating this constraint. The variance computations in (8) and the ReLU specific post-activation do not model any potential covariance structure between the pre-/post-activations units of a layer. While this is in principle feasible, e.g. along the lines of Wu et al. (2019), it leads to an explosion in the required computational cost and memory, hindering the applicability of the approach to deeper nets. Hence, we stick to a diagonal covariance structure throughout, as Wu et al. (2019) have also shown only little test set performance benefit of modeling it.

Appendix D. Further Details on the BLNN Derivations

Derivation of the Moment Matching. For a single data point and the l-th hidden fully-connected layer[5] consisting of K units with an arbitrary activation function $a(\cdot)$, the post-activation layer output is given as $\mathbf{h}^l = a(\mathbf{f}^l)$, where $\mathbf{f}^l = \mathbf{W}^l\mathbf{h}^{l-1}$. The j-th pre-activation output f_j^l is a sum of K terms $f_j^l = \sum_k w_{jk}^l h_k^{l-1}$, which allows us to assume it to be normal distributed via the CLT due to the independence of the individual w_{jk}^l and h_k^{l-1} terms. The mean and the variance of this random variable can be computed as

$$\mathbb{E}\left[f_j^l\right] = \sum_{k=1}^{K} \mathbb{E}\left[w_{jk}^l\right] \mathbb{E}\left[h_k^{l-1}\right], \tag{7}$$

$$\mathrm{var}\left[f_j^l\right] = \sum_{k=1}^{K} \mathbb{E}\left[(w_{jk}^l)^2\right] \mathrm{var}\left[h_k^{l-1}\right] + \mathrm{var}\left[w_{jk}^l\right] \left(\mathbb{E}\left[h_k^{l-1}\right]\right)^2, \tag{8}$$

where we drop any potential covariance structure between the outputs of a layer. The mean and the variance of the weights are readily available via the distributions $p_\phi(\mathbf{w}_n)$. For common activations such as the ReLU, $a(h_k^{l-1}) = \max(0, h_k^{l-1})$, which we will rely on in this work, closed-form solutions to the first two moments of h_k^{l-1} are tractable (Frey and Hinton, 1999) given the moments of the pre-activations of the previous layer $\mathbf{f}^{l-1}$. This gives a recursive scheme terminating at the input layer , where $f_j^1 = \sum_k w_{hj}^1 x_k$. As x_k is a constant, its first moment is itself and the second is zero. Consequently,

$$\mathbb{E}\left[f_j^1\right] = \sum_{k=1}^{K} \mathbb{E}[w_{jk}^1]x_k \quad \text{and} \quad \mathrm{var}\left[f_j^1\right] = \sum_{k=1}^{K} \mathrm{var}[w_{jk}^l]x_k^2,$$

completing the full recipe of how all weights of a BNN can be recursively integrated out from bottom to top, subject to a tight approximation. Scenarios with stochastic input $\mathbf{x} \sim p(\mathbf{x})$ typically entail controllable assumptions on $p(\mathbf{x})$. The equations above remain intact after adding an expectation operator around x_k and x_k^2, readily available for any explicitly defined $p(\mathbf{x})$. Contrarily to the case in GPs, stochastic inputs can be trivially adapted into this framework, greatly simplifying the math for uncertainty-sensitive setups. For a net with L layers, the outcome is a distribution over the final latent hidden layer $\mathbf{f}_n^L \sim \mathcal{N}(\mathbf{f}_n^L|\mathbf{m}_n, \mathbf{s}_n^2)$, simplifying the highly nonlinear integrals in (2) to

$$\int p(\mathbf{y}_n|\mathbf{x}_n, \mathbf{w}_n)p(\mathbf{w}_n)d\mathbf{w}_n \approx \int p(\mathbf{y}_n|\mathbf{f}_n^L)\mathcal{N}(\mathbf{f}_n^L|\mathbf{m}_n, \mathbf{s}_n^2)d\mathbf{f}_n^L,$$

leaving us in a much simpler situation as we can choose a suitable distribution family for $\mathbf{y}_n$.

First two Moments of the ReLU Activation. Mean and variance of a normally distributed variable transformed by the ReLU activation function are analytical tractable (Frey and Hinton, 1999). Following the notation from the main paper, we have that for $h_k^{l-1} = \max(0, f_k^l)$ where $f_k^{l-1} \sim \mathcal{N}(f_k^{l-1}|\mu, \sigma^2)$ for some mean and variance they can be computed as

$$\mathbb{E}\left[h_k^{l-1}\right] = \mathbb{E}_{\mathcal{N}(f_k^{l-1}|\mu,\sigma^2)}\left[\max(0, f_k^{l-1})\right] = \mu\Phi\left(\frac{\mu}{\sigma}\right) + \sigma\phi\left(\frac{\mu}{\sigma}\right),$$

5. Convolutional layers follow analogously.

$$\operatorname{var}\left[h_k^{l-1}\right] = \operatorname{var}\left[\max(0, f_k^{l-1})\right] = (\mu^2+\sigma^2)\Phi\left(\frac{\mu}{\sigma}\right) + \mu\sigma\phi\left(\frac{\mu}{\sigma}\right) - \left(\mathbb{E}\left[h_k^{l-1}\right]\right)^2,$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the cdf and pdf of the standard normal distribution respectively.

First two Moments of the ELU Activation. The following derivations are an adaptation of the ReLU results to ELU in order to scale to deeper networks. We are again interested in the first two moments for the ELU defined as

$$g(x) = \begin{cases} x, & x > 0 \\ \alpha(\exp(x) - 1) & x < 0 \end{cases}. \tag{9}$$

With $f(x) = \max(0, x)$, i.e. the ReLU activation, we have for the expectation of $g(\cdot)$, that

$$\mathbb{E}\left[g(x)\right] = \int_{-\infty}^{0} \alpha(\exp(x)-1)\mathcal{N}(x|\mu,\sigma^2)dx + \mathbb{E}\left[f(x)\right].$$

The first term can be split into

$$\alpha\int_{-\infty}^{0} \exp(x)\mathcal{N}(x|\mu,\sigma^2)dx - \alpha\int_{-\infty}^{0}\mathcal{N}(x|\mu,\sigma^2).$$

We get for these to terms that the first is equal to

$$\alpha\int_{-\infty}^{0} \exp(x)\mathcal{N}(x|\mu,\sigma^2)dx = \alpha\exp(\mu+\sigma^2/2)\Phi\left(-\frac{\mu+\sigma^2}{\sigma}\right)$$

and the second gives

$$\alpha\int_{-\infty}^{0}\mathcal{N}(x|\mu,\sigma^2) = \alpha\Phi\left(-\frac{\mu}{\sigma}\right) = \alpha\left(1-\Phi\left(\frac{\mu}{\sigma}\right)\right).$$

Combining all of this we end up with

$$\mathbb{E}\left[g(x)\right] = \alpha\left(\exp(\mu+\sigma^2/2)\Phi\left(-\frac{\mu+\sigma^2}{\sigma}\right) - \Phi\left(-\frac{\mu}{\sigma}\right)\right) + \mathbb{E}\left[f(x)\right].$$

For the variance we have the general form

$$\operatorname{var}\left[g(x)\right] = \mathbb{E}\left[g(x)^2\right] - \mathbb{E}\left[g(x)\right]^2$$

in which only the second moment is missing. We have that

$$\mathbb{E}\left[g(x)^2\right] = \int_{-\infty}^{0}\alpha^2(\exp(x)-1)^2\mathcal{N}(x|\mu,\sigma^2)dx + \mathbb{E}\left[f(x)^2\right].$$

The first term expands into the following monstrosity

$$\alpha^2\int_{-\infty}^{0}(\exp(2x) - 2\exp(x) + 1)\mathcal{N}(x|\mu,\sigma^2)dx$$

$$
\begin{aligned}
&= \alpha^2 \int_{-\infty}^{-\frac{\mu}{\sigma}} \big(\exp(2\mu + 2\sigma y) - 2\exp(\mu + \sigma y) + 1\big)\phi(y)dy \\
&= \alpha^2 \left(\exp(2\mu + 2\sigma^2)\Phi\left(-\frac{\mu + 2\sigma^2}{\sigma}\right)\right. \\
&\qquad \left. -2\exp(\mu + \sigma^2/2)\Phi\left(-\frac{\mu + \sigma^2}{\sigma}\right) + \Phi\left(-\frac{\mu}{\sigma}\right)\right).
\end{aligned}
$$

These two moments finally can be used as replacements for the ReLU moments in the main paper for deeper networks.

Derivation of the Marginalization for Regression. We approximate the marginal distribution

$$
p(\lambda_n) = \int \mathcal{N}\big(\lambda_n | f_{n1}^L, \exp(f_{n2}^L)\big)\mathcal{N}(\mathbf{f}_n^L | \mathbf{m}_n, \mathbf{s}_n^2)d\mathbf{f}_n^L
$$

with a normal distribution by a further moment matching step. Dropping the indices n and L for notational simplicity, the mean of the right hand side is given as

$$
\begin{aligned}
\mathbb{E}_{p(\lambda)}[\lambda] = \int \lambda p(\lambda)d\lambda &= \int \lambda \mathcal{N}(\lambda | f_1, \exp(f_2))\mathcal{N}(\mathbf{f}|\mathbf{m}, \mathbf{s}^2)d\mathbf{f}d\lambda \\
&= \int f_1 \mathcal{N}(\mathbf{f}|\mathbf{m}, \mathbf{s}^2)d\mathbf{f} = m_1.
\end{aligned}
$$

For the variance term we rely on the law of total variance and have

$$
\begin{aligned}
\mathrm{var}_{p(\lambda)}[\lambda] &= \mathbb{E}_{p(\mathbf{f})}\left[\mathrm{var}_{p(\lambda|\mathbf{f})}[\lambda]\right] + \mathrm{var}_{p(\mathbf{f})}\left[\mathbb{E}_{p(\lambda|\mathbf{f})}[\lambda]\right] \\
&= \mathbb{E}_{p(\mathbf{f})}[\exp(f_2)] + \mathrm{var}_{p(\mathbf{f})}[f_1] \\
&= \int \exp(f_2)\mathcal{N}(f_2|m_2, \sigma_2^2)df_2 + s_1^2 \\
&= \exp(m_2 + s_2^2/2) + s_1^2,
\end{aligned}
$$

where the last integral is given as the mean of a log-normal random variable. Altogether we end up with the desired

$$
p(\lambda_n) \approx \mathcal{N}\Big(\lambda_n \Big| m_{n1}, s_{n1}^2 + \exp(m_{n2} + s_{n2}^2/2)\Big).
$$

This then allows us to compute the log marginal likelihood

$$
\begin{aligned}
\log p(y_n|\mathbf{x}_n) &= \log \int \mathcal{N}(y_n|\lambda_n, \beta^{-1})\left(\int \mathcal{N}\big(\lambda_n|f_{n1}^L, \exp(f_{n2}^L)\big)\mathcal{N}(\mathbf{f}_n^L|\mathbf{m}_n, \mathbf{s}_n^2)d\mathbf{f}_n^L\right)d\lambda_n \\
&\approx \log \int \mathcal{N}(y_n|\lambda_n, \beta^{-1})\mathcal{N}\Big(\lambda_n\Big|m_{n1}, s_{n1}^2 + \exp(m_{n2} + s_{n2}^2/2)\Big)d\lambda_n \\
&= \log \mathcal{N}\Big(y_n\Big|m_{n1}, \beta^{-1} + s_{n1}^2 + \exp(m_{n2} + s_{n2}^2/2)\Big).
\end{aligned}
$$

Appendix E. Derivation of the PAC-Bound

This section gives a more detailed derivation of the individual results stated in the main paper. As stated there, given a predictor h chosen from a hypothesis class $\mathcal{H}$ as a mapping from $\mathbf{x}$ to y, we define the true and the empirical risks as

$$R(h) = -\mathbb{E}_{\mathbf{x},\mathbf{y}\sim\Delta}\left[p\big(\mathbf{y}|h(\mathbf{x})\big)\right], \tag{10}$$

$$R_{\mathcal{D}}(h) = -\frac{1}{N}\sum_{n=1}^{N} p\big(\mathbf{y}_n|h(\mathbf{x}_n)\big) \tag{11}$$

for the data set $\mathcal{D}$ drawn from an arbitrary and unknown data distribution Δ. $R(h)$ and $R_{\mathcal{D}}(h)$ are bounded below by $-\max p(y|h(x))$ and above by zero.

Theorem 2.1 in (Germain et al., 2009) gives us that for any $\delta \in (0,1]$ and any convex function $d(\cdot,\cdot)$

$$\Pr\left\{ d\Big(\mathbb{E}_{h\sim Q}\left[R_{\mathcal{D}}(h)\right], \mathbb{E}_{h\sim Q}\left[R(h)\right]\Big) \leq \frac{\mathrm{KL}\left(Q \parallel P\right) + \log(B/\delta)}{N})\right\} \geq 1-\delta, \tag{12}$$

where $B = \mathbb{E}_{\mathcal{D}\sim\Delta}\left[\mathbb{E}_{h\sim P}\left[\exp\left(Nd(R_{\mathcal{D}}(h), R(h))\right)\right]\right]$. The PAC framework necessitates a convex and non-negative distance measure for risk evaluations. Common practice is to rescale the risk into the unit interval, define the KL divergence as the distance measure, and upper bound its intractable inverse (Germain et al., 2016) using Pinsker's inequality (Catoni, 2007; Dziugaite and Roy, 2017). We follow an alternative path. As our risk is bounded but not restricted to the unit interval, we choose our distance measure as $d(r,r') = (r-r')^2$ and avoid the Pinsker's inequality step.

Adapting the standard KL inversion trick (Seeger, 2002) to the Euclidean distance, we can simply define $d^{-1}(x,\varepsilon) = \max\{x' : (x-x')^2 = \varepsilon\} = x + \sqrt{\varepsilon}$ for some $\varepsilon \geq 0$. We apply this function to both sides of the inequality and get

$$\begin{aligned} & d^{-1}\Big(\mathbb{E}_{h\sim Q}\left[R_{\mathcal{D}}(h)\right], d\big(\mathbb{E}_{h\sim Q}\left[R_{\mathcal{D}}(h)\right], \mathbb{E}_{h\sim Q}\left[R(h)\right]\big)\Big) \\ & \leq d^{-1}\Big(\mathbb{E}_{h\sim Q}\left[R_{\mathcal{D}}(h)\right], \big(KL(Q||P) + \log(B/\delta)\big)/N\Big), \end{aligned}$$

where $d(\cdot,\cdot) \geq 0$ and $KL(Q||P) \geq 0$ by definition and because $\delta \in [0,1]$ and $e^{d(\cdot,\cdot)} \geq 0$ we have $\log(B/\delta) = -\log\delta + \log B \geq 0$. Since

$$\mathbb{E}_{h\sim Q}\left[R(h)\right] \leq \mathbb{E}_{h\sim Q}\left[R(h)\right] + d\Big(\mathbb{E}_{h\sim Q}\left[R_{\mathcal{D}}(h)\right], \mathbb{E}_{h\sim Q}\left[R(h)\right]\Big)$$

directly follows from $d^{-1}(\cdot,\cdot)$, we bound the true risk as

$$\Pr\left\{ \mathbb{E}_{h\sim Q}\left[R(h)\right] \leq \mathbb{E}_{h\sim Q}\left[R_{\mathcal{D}}(h)\right] + \sqrt{\frac{KL(Q||P) + \log(B/\delta)}{N}} \right\} \geq 1-\delta.$$

This outcome has a similar structure to application of Pinsker's inequality to a setup with risk defined on the unit interval, but without such a restriction. Hence, the implied upper bound is no longer trivial. To arrive at the final bound we have to further approximate each of the two terms of the bound.

For the first term, we have that

$$\begin{aligned}\mathbb{E}_{h\sim Q}\left[R_{\mathcal{D}}(h)\right] &= -\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}_{h\sim Q}\left[p\big(\mathbf{y}_n|h(\mathbf{x}_n)\big)\right]\\ &\leq -\frac{1}{N}\sum_{n=1}^{N}\log\mathbb{E}_{h\sim Q}\left[p\big(\mathbf{y}_n|h(\mathbf{x}_n)\big)\right]\\ &= -\frac{1}{N}\sum_{n=1}^{N}\log p(\mathbf{y}_n|\mathbf{x}_n),\end{aligned}$$

where the inequality uses that $-\log(u) > -u \; \forall u \in (0,\infty)$ and the equality follows by the marginalisation techniques discussed in the main paper.

To get a tractable second term we process B further. Exploiting the fact that

$$\mathbb{E}_{x\sim p(x)}\left[\max f(x)\right] \leq \max f(x) \tag{13}$$

for any $p(\cdot)$ and $f(\cdot)$, we can drop the the expectation term and get

$$\begin{aligned}B &= \mathbb{E}_{\mathcal{D}\sim\Delta}\left[\mathbb{E}_{h\sim P}\left[e^{N(R_{\mathcal{D}}(h)-R(h))^2}\right]\right]\\ &\leq \max e^{N(R_{\mathcal{D}}(h)-R(h))^2}.\end{aligned} \tag{14}$$

For a multiclass classification, the likelihood is bounded into the interval $[0,1]$ such that with $\max(R_{\mathcal{D}}(h)-R(h))^2 = (0-1)^2 = 1$ we have that

$$B \leq e^N \quad \Rightarrow \quad \log B \leq N. \tag{15}$$

For regression, with the likelihood $\mathcal{N}(y|h(x),\beta^{-1})$, we have that $R_{\mathcal{D}}(h)$ and $R(h)$ are bounded from above by 0 and from below by $-\mathcal{N}\big(y=\mu|\mu,\beta^{-1}\big)$, i.e. by the density at the mode of a normal distribution with precision β^{-1}. Hence,

$$B \leq e^{N\left(0-\mathcal{N}(\mu,\beta^{-1})\right)^2} \quad \Rightarrow \quad \log B \leq N\frac{\beta}{2\pi}. \tag{16}$$

Combining these relaxations we get the objectives described in the main paper.

Appendix F. Experimental Details and Further Experiments

This section contains further experiments as well as details on the hyperparameters of the experiments performed in the main paper.

F.1. Regression

We evaluate the regression performance of BEDL+Reg and the baselines on eight standard UCI benchmark data sets. Adopting the experiment protocol introduced in (Hernández-Lobato and Adams, 2015), we use 20 random train-test set splits comprising 90% and

N/d	`boston` 506/13	`concrete` 1030/8	`energy` 768/8	`kin8nm` 8192/8	`naval` 11934/16	`power` 9568/4	`protein` 45730/9	`wine` 1599/11
Sparse GP	-2.22 ± 0.07	-2.85 ± 0.02	-1.29 ± 0.01	1.31 ± 0.01	4.86 ± 0.04	-2.66 ± 0.01	-2.95 ± 0.05	-0.67 ± 0.01
MC Dropout	-2.46 ± 0.25	-3.04 ± 0.09	-1.99 ± 0.09	0.95 ± 0.03	3.80 ± 0.05	-2.89 ± 0.01	-2.80 ± 0.05	-0.93 ± 0.06
VarOut	-2.63 ± 0.02	-3.15 ± 0.02	-3.29 ± 0.00	1.09 ± 0.01	5.50 ± 0.03	-2.82 ± 0.01	-2.90 ± 0.01	$\mathbf{-0.88 \pm 0.02}$
PBP	-2.57 ± 0.09	-3.16 ± 0.02	-2.04 ± 0.02	0.90 ± 0.01	3.73 ± 0.01	-2.84 ± 0.01	-2.97 ± 0.00	-0.97 ± 0.01
DVI	$\mathbf{-2.41 \pm 0.02}$	-3.06 ± 0.01	-1.01 ± 0.06	1.13 ± 0.00	$\mathbf{6.29 \pm 0.04}$	-2.80 ± 0.00	-2.84 ± 0.01	-0.90 ± 0.01
BEDL-Hyper (Ours)	-2.57 ± 0.04	-3.30 ± 0.01	-2.59 ± 0.02	0.44 ± 0.00	3.69 ± 0.00	-2.98 ± 0.01	-3.00 ± 0.00	-1.00 ± 0.01
BEDL (Ours)	-2.45 ± 0.08	-3.09 ± 0.06	-0.87 ± 0.10	1.12 ± 0.01	5.76 ± 0.07	-2.80 ± 0.01	-2.82 ± 0.01	-0.93 ± 0.01
BEDL+Reg (Ours)	-2.43 ± 0.06	$\mathbf{-3.02 \pm 0.02}$	$\mathbf{-0.73 \pm 0.04}$	$\mathbf{1.15 \pm 0.01}$	5.60 ± 0.11	$\mathbf{-2.79 \pm 0.01}$	$\mathbf{-2.77 \pm 0.01}$	-0.90 ± 0.01

Table 2: **Regression.** Average test log-likelihood $\pm$ standard error over 20 random train/test splits. N/d give the number of data points in the complete data set and the number of input feature. The sparse GP results are cited from (Bui et al., 2016) and VarOut relies on our own implementation.

10% of the samples, respectively. The nets consist of a single hidden layer with 50 units and ReLU nonlinearities.[6] The hypothesis class in this task is over the regularization parameters $P := p(\boldsymbol{\lambda}) = \prod_n \mathcal{N}(\lambda_n|0, \alpha^{-1})$, with precision α, while Q is given as $Q := p(\boldsymbol{\lambda}) = \prod_n \int p(\lambda_n|\mathbf{f}_n)p(\mathbf{f}_n)d\mathbf{f}_n$. We compare BEDL+Reg against the state of the art in BNN inference methods that do not require sampling across neural net weights, *Probabilistic Back-Propagation* (PBP) (Hernández-Lobato and Adams, 2015) and *Deterministic Variational Inference* (DVI) (Wu et al., 2019), which use the CLT-based moment matching for expectation propagation and VI, respectively. For completeness, we also compare against the two most common sampling-based alternatives, *Variational Dropout* (VarOut) (Kingma et al., 2015; Molchanov et al., 2017) and *MC Dropout* (Gal and Ghahramani, 2016). In the results summarized in Table 2, BEDL+Reg outperforms all baselines in the majority of the data sets and is competitive in the others.

The PAC regularization improves over BEDL in all data sets except one. We also report results for a sparse GP with 50 inducing points, which approximates a BNN of one infinitely wide hidden layer (Neal, 1995). As expected, the GP sets a theoretical upper bound on BEDL+Reg as well as the baselines for one hidden layer architectures. Lastly, we compare our tediously derived PAC regularizer to straightforward Maximum-A-Posteriori estimation on the BEDL hyperpriors (BEDL-Hyper) (see the appendix for details), which deteriorates performance on all UCI data sets.

Experimental Setup. The neural net used consists of a single hidden layer of 50 units for all data sets except `protein`, which gets 100. The results for all of the baselines except for Variational Dropout (VarOut) are quoted from the results reported by the respective papers who introduced them, while the results on the sparse GP are reported via (Bui et al., 2016). For VarOut we rely on our own implementation as there are no official results. BEDL, BEDL+Reg, and VarOut all share the same initialization scheme for the mean and variance parameters for each weight following the initialization of (Louizos and Welling, 2017), i.e. He-Normal for the means and $\mathcal{N}(-9, 0.001)$ for the log variances. VarOut gets a Normal prior with a precision of $\alpha = 1.0$, and all three get an observation precision of $\beta = 100$, to encourage them to learn as much of the predictive uncertainty instead of relying on a fixed

6. Except for the larger `protein`, which gets 100 hidden units.

hyper-parameter. Note that we keep these values fix and data set independent, different to many of the baselines who set them to data set specific values given cross-validations on separate validation subsets.

Each model is trained with the Adam optimizer with default parameters for 100 epochs with a learning rate of 10^{-3}, with varying minibatch sizes depending on the data set size.

F.2. Classification and Out-of-domain Detection.

The network for this task follows the common LeNet5 architecture with the following modifications. Instead of max-pooling layers after the two convolutional layers, the convolutional layers themselves use a larger stride to mimic the behavior. And for the more complex CIFAR data set the number of channels in the two convolutional layers is increased from the default 20,50 to 192 each, while the number of hidden units for the fully connected layer is increased from 500 to 1000 for that data set following (Gal and Ghahramani, 2015).

Since there are no OOD results on the BNN baselines we compare against, we rely on our own reimplementations of them, ensuring that they each share the same initialization schemes as in the regression setup. For DVI we implement the diagonal version and use a sampling-based approximation on the intractable softmax. Each model gets access to five samples whenever it needs to conduct an MC sampling approximation. All models get trained via the Adam optimizer with the default hyperparameters and a learning rate of 10^{-3}. For EDL we rely on the public implementation the authors (Sensoy et al., 2018) provide and use their hyperparameters to learn the model. Due to the license status of the not-MNIST data conflicting with the affiliation of the authors, we have to change the setup of earlier work, e.g. (Lakshminarayanan et al., 2017; Louizos and Welling, 2017; Sensoy et al., 2018), using instead Fashion-MNIST as the closest substitute.

F.3. Comparison to GP variants.

We evaluate the impact of local weight realization on prediction performance by comparing BEDL+Reg to GPs with kernels derived from BNNs with global weight realizations (Garragia-Alonso et al., 2019; Lee et al., 2018; Neal, 1995) on MNIST and CIFAR10 data sets. It is technically not possible to perform this evaluation in a fully commensurate setup, as these baselines assume infinitely many neurons per layer and do not have weight-specific degrees of freedom. Furthermore, Garragia-Alonso et al. (2019) perform neural architecture search and Lee et al. (2018) use only part of the CIFAR10 training set reporting that the rest does not fit into the memory of a powerful workstation. We nevertheless view the performance scores reported in these papers as practical upper bounds and provide qualitative comparison. For the choice of neural net depth, we take NNGP (Lee et al., 2018) as a reference and devise a contrarily thin two-layer convolutional BEDL+Reg network. The results and the architectural details are summarized in Table 3. BEDL and BEDL+Reg can reach lower error rates using significantly less computational resources.

Experimental Setup. The results for the baselines are taken from the respective original papers. The nets for BEDL and BEDL+Reg consist of two convolutional layers with 96 filters of size 5×5 and a stride of 5. They are trained until convergence (50 epochs) using Adam with the default hyperparameters and a learning rate of 10^{-3}.

	MNIST	**CIFAR10**
NNGP	1.21	44.3
Convolutional GP	1.17	35.4
ConvNet GP	1.03	-
Residual CNN GP	0.96	-
ResNet GP	0.84	-
BEDL (Ours)	0.91	34.20
BEDL+Reg (Ours)	**0.63**	**32.47**

Table 3: **Comparison to GP Variants.** Test error in % on two image classification tasks. BEDL reaches lower error rate than previously proposed neural net based GP constructions by two convolutional layers with 96 filters of size 5×5 and stride 2. BEDL converges in 50 epochs, amounting to circa 30 minutes of training time on a single GPU. The GP alternatives have been reported to have significantly larger time and memory requirements. The GP results are cited from (Garragia-Alonso et al., 2019)

F.4. Computational cost.

Table 4 summarizes the computational cost analysis of the considered approaches. MC Dropout and VarOut can quantify uncertainty only by taking samples across weights, which increases the prediction cost linearly to the sample count. DVI and BEDL+Reg perform the forward pass during both training and prediction time via analytical moment matching at double and triple costs, respectively. Both methods have sampling costs for intractable likelihoods.[7] BEDL+Reg may also have another additive per-data-point sampling cost for calculating intractable functional mapping regularizers. Favorably, both of these overheads are only additive to the forward pass cost, i.e. sampling time is independent of the neural net depth, hence they do not set a computational bottleneck. The training and prediction cost of BEDL+Reg is three times EDL which builds on deterministic neural nets. However, it provides substantial improvements in both prediction accuracy and uncertainty quantification.

7. Even this sampling step could be avoided by a suitable Taylor approximation, see e.g. Appendix B.4/B.5 in (Wu et al., 2019). As the added approximation error was more detrimental to model performance than a cheap MC approach in preliminary experiments, we stay with the latter for both.

	Training per iteration	**Prediction**
MC Dropout	$\mathcal{O}\big((F+L)S\big)$	$\mathcal{O}\big((F+L)S\big)$
VarOut	$\mathcal{O}\big(2(F+L)S+R(W/N)\big)$	$\mathcal{O}\big(2(F+L)S\big)$
DVI	$\mathcal{O}\big(2F+SL+R(W/N)\big)$	$\mathcal{O}\big(2F+SL\big)$
EDL	$\mathcal{O}\big(F+L\big)$	$\mathcal{O}\big(F+L\big)$
BEDL	$\mathcal{O}\big(3F+SL\big)$	$\mathcal{O}\big(3F+SL\big)$
BEDL+Reg	$\mathcal{O}\big(3F+S(L+R)\big)$	$\mathcal{O}\big(3F+SL\big)$

Table 4: **Computational Cost.** Per data point computational cost analysis in FLOPs. **F:** Forward pass cost of a deterministic net. **W:** Number of weights in the net. **L:** Analytical calculation cost for the exact or approximate likelihood or the loss term. **S:** Number of samples taken. **R:** The cost of the regularization term per unit (weight or data point).