Make Me a BNN: A Simple Strategy for Estimating Bayesian Uncertainty from Pre-trained Models

Gianni FRANCHI,[†] Olivier Laurent, Maxence Leguery, Andrei Bursuc, Andrea Pilzer, Angela Yao

[†] *gianni.franchi@ensta-paris.fr*

CVPR 2024

Why Quantify Uncertainty in Deep Neural Networks?

Context

- Deep Neural Networks (DNNs) have achieved remarkable success in various applications, but their predictions are not infallible.
- Recognizing and quantifying uncertainty is crucial for enhancing the reliability and trustworthiness of DNNs.

Motivation

- **Real-world Consequences:** In critical applications such as healthcare or autonomous systems, incorrect predictions can have severe consequences.
- **Decision-Making:** Users and decision-makers need to understand the confidence levels associated with DNN predictions.

Types of Uncertainty in Machine Learning

Aleatoric Uncertainty

- **Data Uncertainty:** Arises from inherent variability in the data. It can be further classified into homoscedastic (constant variance) and heteroscedastic (varying variance) uncertainty.
- **Measurement Uncertainty:** Associated with errors in the measurement process, impacting the reliability of observed data.

Epistemic Uncertainty

- **Model Uncertainty:** Arises from a lack of knowledge about the true model structure. It can be reduced with more data and better model architecture.
- **Inherent Model Limitations:** Uncertainty arising from the inability of the model to capture all relevant aspects of the underlying data distribution.
- **Parameter Uncertainty:** Related to uncertainty in the values of model parameters, often addressed through techniques like Bayesian modeling.

Single Network Methods

Notations

- We consider that we have a training dataset $\mathcal{D} := \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\} \subset \mathcal{X} \times \mathcal{Y}$,
- $(\boldsymbol{x}_i, y_i)$ are assumed i.i.d. according to some unknown probability measure $P_{\mathcal{X}\times\mathcal{Y}}$ on $\mathcal{X} \times \mathcal{Y}$
- We denote $f_\omega(\boldsymbol{x})$ the prediction a DNN model with weights ω. We consider that $f_\omega(\boldsymbol{x}) = P(y|\boldsymbol{x}, \boldsymbol{\omega})$

Maximum Likelihood Estimation for Classification

Our goal is to find $\boldsymbol{\omega}$ that maximizes the Likelihood $P(\mathcal{D}|\boldsymbol{\omega})$.
Let us consider the case of i.i.d. samples from the conditional distribution. Then, we can write the likelihood function of $\boldsymbol{\omega}$:

$$\boldsymbol{\omega} = \arg\max_{\boldsymbol{\omega}} P(\mathcal{D}|\boldsymbol{\omega}) \approx \arg\max_{\boldsymbol{\omega}} \sum_{i=1}^{N} \log P(y_i|\boldsymbol{x}_i, \boldsymbol{\omega}) \tag{1}$$

Bayesian Deep Neural Networks [1]

Bayesian DNNs are based on marginalization rather than MAP optim.:

$$P(y|\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{\omega} \sim P(\boldsymbol{\omega}|\mathcal{D})} [P(y|\boldsymbol{x}, \boldsymbol{\omega})] \tag{2}$$

$$P(y|\boldsymbol{x}) = \int P(y|\boldsymbol{x}, \boldsymbol{\omega}) P(\boldsymbol{\omega}|\mathcal{D}) d\boldsymbol{\omega} \tag{3}$$

In practice:

$$P(y|\boldsymbol{x}) \simeq \sum_i P(Y|X, \boldsymbol{\omega}_i), \text{ with } \boldsymbol{\omega}_i \sim P(\boldsymbol{\omega}|\mathcal{D}) \tag{4}$$

$\Rightarrow$ Different methods to estimate $P(\boldsymbol{\omega}|\mathcal{D})$.

Posterior "Landscape" and Ensembles

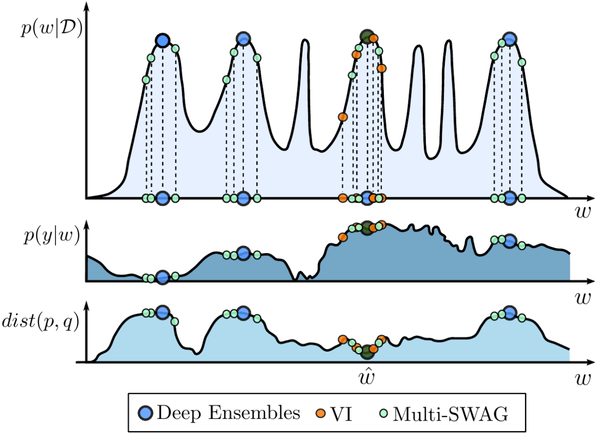

Figure: Top: $P(\omega|\mathcal{D})$, with representations from VI (orange), deep ensembles (blue), multiBNN (red). Middle $P(y|\boldsymbol{x}, \omega)$ (from Wilson & Izmailov [15])

How to estimate the Posterior of BNN?

Classical VI-BNN

Using the "reparametrization trick", a layer j of an MLP can be written:

$$\begin{aligned} \boldsymbol{u}_j &= \text{norm}\left(\left[W_\mu^{(j)} + \epsilon_j W_\sigma^{(j)}\right] \boldsymbol{h}_{j-1}, \beta_j, \gamma_j\right), \text{ and} \\ \boldsymbol{a}_j &= a(\boldsymbol{u}_j), \end{aligned} \tag{5}$$

where the matrices $W_\mu^{(j)}$ and $W_\sigma^{(j)}$ denote the mean and standard deviation of the posterior distribution of layer j, $\epsilon_j \sim \mathcal{N}(0, \mathbb{1})$ and the operator $\text{norm}(\cdot, \beta_j, \gamma_j)$, of trainable parameters β_j and γ_j, can refer to any batch, layer, or instance normalization.

How to turn a DNN into a BNN?

ABNN

Our objective differs from VI-BNN, which requires training the posterior distribution parameters from scratch. Instead, our approach entails leveraging and converting an existing DNN into a BNN.

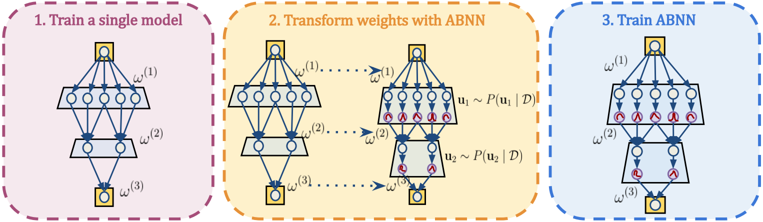

Figure: **Illustration of the training process for the ABNN.** The procedure begins with training a single DNN $\boldsymbol{\omega}_{\mathrm{MAP}}$, followed by architectural adjustments to transform it into an ABNN. The final step involves fine-tuning the ABNN model.

How to turn a DNN into a BNN?

ABNN

Formally, our BNN relies on a new layer $\textbf{BNL}(\cdot)$:

$$\begin{aligned} \boldsymbol{u}_j &= \textbf{BNL}\left(W^{(j)}\boldsymbol{h}_{j-1}\right), \text{ and} \boldsymbol{a}_j = a(\boldsymbol{u}_j), \text{ with} \\ \textbf{BNL}(\boldsymbol{h}_j) &= \frac{\boldsymbol{h}_j - \hat{\mu}_j}{\hat{\sigma}_j} \times \gamma_j(1+\epsilon_j) + \beta_j. \end{aligned} \tag{6}$$

This can be seen as adding a Gaussian dropout on normalization layer and finetuning the DNN. We propose to train multiple of these ABNNs to have multiple modes of the posterior.

ABNN during evaluation

During evalution, for each sample from ABNN $\boldsymbol{\omega}_m$, we augment the number of samples by independently sampling multiple $\epsilon_j \sim \mathcal{N}(0, \mathbb{1})$.

$$P(y \mid \boldsymbol{x}, \mathcal{D}) \approx \frac{1}{ML} \sum_{l=1}^{L} \sum_{m=1}^{M} P(y \mid \boldsymbol{x}, \boldsymbol{\omega}_m, \epsilon_l). \tag{7}$$

Classification Results

	Method	CIFAR-10 Acc ↑	CIFAR-10 NLL ↓	CIFAR-10 AUPR ↑	CIFAR-10 AUC ↑	CIFAR-10 FPR95 ↓	CIFAR-100 Acc ↑	CIFAR-100 NLL ↓	CIFAR-100 AUPR ↑	CIFAR-100 AUC ↑	CIFAR-100 FPR95 ↓	Time (h) ↓
ResNet-50	Single Model	95.1	0.211	95.2	91.9	23.6	78.3	0.905	87.4	77.9	57.6	**1.7**
	BatchEnsemble	93.9	0.255	94.7	91.3	20.1	66.6	1.788	85.2	74.6	60.6	17.2
	LPBNN	94.3	0.231	92.7	86.7	54.9	78.5	1.02	88.2	77.8	73.5	17.2
	MCDropout	94.4	0.190	93.1	86.9	43.8	76.9	0.858	87.8	77.1	64.1	**1.7**
	MCBN	95.0	0.168	95.7	92.6	20.1	78.4	0.83	86.8	77.5	57.7	**1.7**
	Deep Ensembles	**96.0**	**0.136**	**97.0**	**94.7**	**80.9**	**0.713**	**2.6**	89.2	80.8	52.5	6.8
	Laplace	95.3	0.160	96.0	93.3	78.2	0.99	14.2	89.2	81.0	51.8	**1.7**
	ABNN	95.0	0.160	96.5	93.9	17.5	77.8	0.828	**90.0**	**82.0**	**51.3**	2.0
WideResNet-28×10	Single Model	95.4	0.200	96.1	93.2	20.4	80.3	0.963	81.0	64.2	80.1	**4.2**
	BatchEnsemble	95.6	0.206	95.5	92.5	22.1	82.3	0.835	88.1	78.2	69.8	25.6
	LPBNN	95.1	0.249	95.4	91.2	29.5	79.7	0.831	79.0	70.1	71.4	23.3
	MCDropout	95.7	0.138	96.2	93.5	12.8	79.2	0.758	**89.4**	**80.1**	58.6	**4.2**
	MCBN	95.5	**0.133**	96.5	94.2	14.6	80.4	0.749	80.4	67.8	63.1	**4.2**
	Deep Ensembles	**95.8**	0.143	**97.8**	**96.0**	**82.5**	0.903	22.9	81.6	67.9	71.3	16.6
	Laplace	95.6	0.151	95.0	90.7	31.9	80.1	0.942	83.4	72.1	59.9	**4.2**
	ABNN	94.5	0.171	0.7	96.8	94.6	80.0	**0.734**	86.7	75.7	59.4	5.0

→ ABNN improves uncertainty quantification with small computational overhead

→ Most of the gains are linked to improved epistemic uncertainty (as measured by OOD detection)

Semantic segmentation Results

	Method	mIoU ↑	ECE ↓	AUPR ↑	AUC ↑	FPR95 ↓
StreetHazards	Single Model	53.9	6.5	6.9	86.6	35.7
	TRADI	52.5	6.3	6.9	87.4	38.3
	Deep Ensembles	55.6	5.3	**8.3**	87.9	**30.3**
	BatchEnsemble	**56.2**	6.1	7.6	88.2	32.9
	LP-BNN	54.5	**5.2**	7.2	88.3	32.6
	ABNN	53.8	6.1	7.9	**88.4**	32.0
BDD-Anomaly	Single Model	47.6	17.7	4.5	85.2	28.8
	TRADI	44.3	16.6	4.5	84.8	36.9
	Deep Ensembles	**51.1**	14.2	5.2	84.8	**28.6**
	BatchEnsemble	48.1	16.9	4.5	84.3	30.2
	LP-BNN	49.0	17.2	4.5	85.3	29.5
	ABNN	48.8	**14.0**	**6.0**	**85.7**	29.0
MUAD	Single Model	57.3	6.1	26.0	86.2	39.4
	MC-Dropout	55.6	6.5	22.3	84.4	45.8
	Deep Ensembles	58.3	5.9	**28.0**	87.1	37.6
	BatchEnsemble	57.1	6.0	25.7	86.9	38.8
	ABNN	**62.0**	**5.6**	24.4	**91.6**	**21.7**

→ ABNN also performs well in the segmentation setting

Conclusions

Exploring Further

- **Contribute to Torch Uncertainty:** If you are interested in advancing the field, consider contributing to TorchUncertainty.

  https://github.com/ENSTA-U2IS-AI/torch-uncertainty

- **Explore Our Resources:** Check out our curated list of resources on Uncertainty, available at our "awesome of uncertainty" repository.

  https://github.com/ENSTA-U2IS-AI/awesome-uncertainty-deeplearning

Bibliography:

1 Blundell, Charles, et al. "Weight uncertainty in neural networks." arXiv preprint arXiv:1505.05424 (2015)

2 A.G. Wilson, P. Izmailov. Bayesian Deep Learning and a Probabilistic Perspective of Generalization. Advances in Neural Information Processing Systems, 2020.

3 Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell. "Simple and scalable predictive uncertainty estimation using deep ensembles." Advances in neural information processing systems. 2017.

4 Gal, Yarin, and Zoubin Ghahramani. "Dropout as a bayesian approximation: Representing model uncertainty in deep learning." international conference on machine learning. 2016.

Bibliography:

5 A.G. Wilson, P. Izmailov. Bayesian Deep Learning and a Probabilistic Perspective of Generalization. Advances in Neural Information Processing Systems, 2020.

6 Wen, Yeming, Dustin Tran, and Jimmy Ba. "Batchensemble: an alternative approach to efficient ensemble and lifelong learning." arXiv preprint arXiv:2002.06715 (2020).

7 Franchi, G., Yu, X., Bursuc, A., Tena, A., Kazmierczak, R., Dubuisson, S., ... & Filliat, D. (2022). MUAD: Multiple Uncertainties for Autonomous Driving, a benchmark for multiple uncertainty types and tasks. arXiv preprint arXiv:2203.01437.

8 Gawlikowski, J., Tassi, C.R.N., Ali, M., Lee, J., Humt, M., Feng, J., Kruspe, A., Triebel, R., Jung, P., Roscher, R. and Shahzad, M., 2023. A survey of uncertainty in deep neural networks. Artificial Intelligence Review, 56(Suppl 1), pp.1513-1589.

Bibliography:

9 Zhang, R., Li, C., Zhang, J., Chen, C., & Wilson, A. G. (2019). Cyclical stochastic gradient MCMC for Bayesian deep learning. arXiv preprint arXiv:1902.03932.

10 Franchi, G., Bursuc, A., Aldea, E., Dubuisson, S., & Bloch, I. (2020). TRADI: Tracking deep neural network weight distributions. In ECCV 2020.