Encoding the latent posterior of Bayesian Neural Networks for uncertainty quantification

Gianni Franchi[1], Andrei Bursuc[2], Emanuel Aldea[3], Séverine Dubuisson[4] and Isabelle Bloch[5]
1: ENSTA Paris, Institut polytechnique de Paris, 2: valeo.ai, 3: SATIE, Université Paris-Saclay, 4: CNRS, LIS, Aix Marseille University, 5: Télécom Paris, Institut polytechnique de Paris

Context and objectives

Here we propose to turn **Bayesian neural networks (BNNs)** [Blu+15] scalable.

- In this work we aim for **efficient** deep BNNs amenable to complex **computer vision** architectures, e.g. ResNet50 [He+16] DeepLabV3+ [Che+18].
- We achieve this thanks to **variational autoencoders (VAEs)** [KW14] .
- Our approach, **Latent-Posterior BNN (LP-BNN)**, is compatible with the recent BatchEnsemble [WTB20] method.
- Our approach is efficient and has results close to the **state of the art**.

The code can be downloaded in `https://github.com/giannifranchi/LP_BNN`

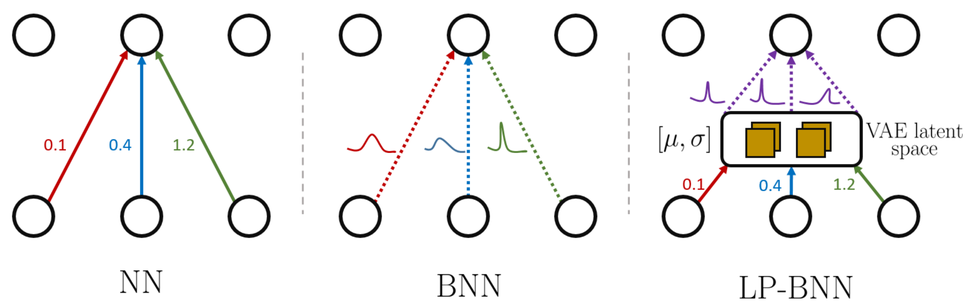

Fig. 1: Illustration of a standard DNN a standard BNN and LP-BNN.

Deep Neural Network (DNN) and Uncertainty

- **BNNs** [Blu+15]: aim to find the posterior distribution of the parameters given the training dataset $P(\Theta \mid \mathcal{D})$, not only the values corresponding to the MAP. To make a prediction y on a new sample $\mathbf{x}$ the BNN computes :

$$P(y \mid \mathbf{x}, \mathcal{D}) = \int P(y \mid \mathbf{x}, \Theta) P(\Theta \mid \mathcal{D}) d\Theta,$$

- **Deep Ensembles**[LPB17]: train mutliple DNNs to have access to their uncertainty.

- **BatchEnsemble** [WTB20]: builds an ensemble from a single base network. Each layer is composed of *"slow weights"* (W_{share}) shared among all data of one batch, and a Rank-1 matrix that varies among all batch data, called *"fast weight"* ($\{\mathbf{s}_j, \mathbf{r}_j\}_{j=1}^{J}$).

$$h = a\left((W_{\text{share}}^{\top}(\mathbf{x} \odot \mathbf{s}_j)) \odot \mathbf{r}_j\right),$$

where a is an activation function and h the output activations.

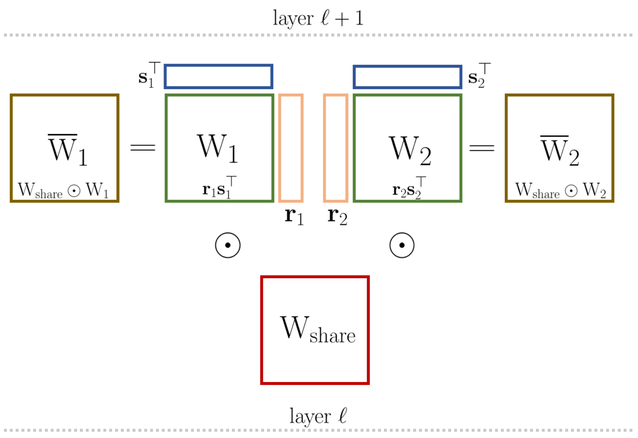

Fig. 2: Illustration on how BatchEnsemble generates the ensemble weights for an ensemble of size $J = 2$.

LP-BNN

- We propose to compute the **posterior distribution of the fast weights** instead of learning the posterior distribution of the weight.
- This can be efficiently done with a **VAE** [KW14] that can find a variational approximation $Q_\phi(\mathbf{z} \mid \mathbf{r})$ to the intractable posterior $P_\psi(\mathbf{z} \mid \mathbf{r})$.
- For each layer of the network $f_\Theta(\cdot)$ we introduce a **VAE composed of a one layer encoder** $g_\phi^{\text{enc}}(\cdot)$ with variational parameters ϕ and **a one layer decoder** $g_\psi^{\text{dec}}(\cdot)$ with parameters ψ.
- The prior over the latent variables is a centered isotropic Gaussian $P_\psi(\mathbf{z}) = \mathcal{N}(\mathbf{z}; 0, \mathbf{I})$ (we adopt the commonly used approach)
- The encoder takes as input a mini-batch of size J (the size of the ensemble) composed of all the $\mathbf{r}_j$ weights of this layer and outputs as activations $(\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j^2)$.
- At each forward pass, **we sample new fast weights $\hat{\mathbf{r}}_j$ from the latent posterior** distribution to be further used for generating the ensemble.
- The BNN is trained in the standard manner with the ELBO loss [KW14]:

$$\mathcal{L}_{\text{LP-BNN}}(\Theta^{\text{LP-BNN}}) = -\sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}} \mathbb{E}_{\mathbf{z} \sim Q_\phi(\mathbf{z} \mid \mathbf{r})} \log\left(P(y_i \mid \mathbf{x}_i, \Theta^{\text{LP-BNN}}, \mathbf{z})\right) + \mathrm{KL}(Q_\phi(\mathbf{z} \mid \mathbf{r}) || P_\psi(\mathbf{z})) + \|\mathbf{r} - \hat{\mathbf{r}}\|^2,$$

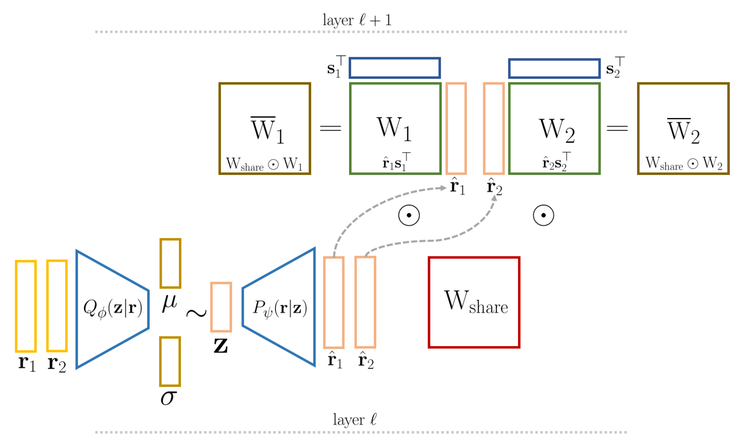

Fig. 3: Illustration on how LP-BNN generates the ensemble weights ($J = 2$).

Experimental results

We evaluate the performance of LP-BNN in assessing the uncertainty of its predictions on **CIFAR-10/100** [KH+09] **StreetHazards** [Hen+19] and **BDD-Anomaly**[Hen+19].
We notice that **DE with cutout outperforms others on most of the metrics** except ECE, cA, and cE on CIFAR-10 , and cA on CIFAR-100, where LP-BNN achieves state of the art results. This means that **LP-BNN is competitive for aleatoric uncertainty estimation**. In fact, ECE is calculated on the test set of CIFAR-10 and CIFAR-100, so it mostly measures the reliability of the confidence score in the training distribution. cA and cE are evaluated on corrupted versions of CIFAR-10 and CIFAR-100 [HD18], which amounts to quantifying the aleatoric uncertainty. On the other hand, for epistemic uncertainty, we can see that DE always attain best results. Yet, **LP-BNN, in most cases, performs close to DE**. Computation wise, **DE takes** 52 **hours** to train on CIFAR-10, while **our solution needs 2 times less,** 26 **hours and** 30 **minutes**. Overall, our LP-BNN is more computationally efficient while providing better results for the aleatoric uncertainty.

Method	CIFAR-10 Acc ↑	AUC ↑	AUPR ↑	FPR-95-TPR ↓	ECE ↓	cA ↑	cE ↓	CIFAR-100 Acc ↑	ECE ↓	cA ↑	cE ↓
MCP + cutout	96.33	0.9600	0.9767	0.115	0.0207	32.98	0.6167	80.19	0.1228	19.33	0.7844
MC dropout	95.95	0.9126	0.9511	0.282	0.0172	32.32	0.6673	75.40	0.0694	19.33	0.5830
MC dropout +cutout	96.50	0.9273	0.9603	0.242	0.0117	32.35	0.6403	77.92	0.0572	27.66	0.5909
Deep Ensembles + cutout	**96.74**	**0.9803**	**0.9896**	**0.071**	**0.0093**	68.75	0.1414	**82.29**	**0.0524**	47.35	**0.1981**
BatchEnsembles + cutout	96.48	0.9540	0.9731	0.132	0.0167	**71.67**	0.1928	81.27	0.0912	47.44	0.2909
LP-BNN (ours) + cutout	95.02	**0.9691**	**0.9836**	**0.103**	**0.0094**	**69.51**	**0.1197**	76.85	0.0677	**47.80**	0.2324

Tab. 1: Comparative results for classification tasks on CIFAR-10 and CIFAR-100. The results are averaged over three seeds.

Dataset	OOD method	mIoU ↑	AUC ↑	AUPR ↑	FPR-95-TPR ↓	ECE ↓
StreetHazards DeepLabv3+ ResNet50	Baseline (MCP)	53.90	0.8660	0.0691	0.3574	0.0652
	TRADI	52.46	0.8739	0.0693	0.3826	0.0633
	Deep Ensembles	55.59	0.8794	**0.0832**	0.3029	0.0533
	BatchEnsemble	**56.16**	0.8817	0.0759	0.3285	0.0609
	LP-BNN (ours)	54.50	**0.8833**	0.0718	0.3261	**0.0520**
	LP-BNN + GN (ours)	**56.12**	**0.8908**	0.0742	**0.2999**	**0.0593**
BDD-Anomaly DeepLabv3+ ResNet50	Baseline (MCP)	47.63	0.8515	0.0450	0.2878	0.1768
	TRADI	44.26	0.8480	0.0454	0.3687	0.1661
	Deep Ensembles	**51.07**	0.8480	0.0524	**0.2855**	**0.1419**
	BatchEnsemble	48.09	0.8427	0.0449	0.3017	0.1690
	LP-BNN (ours)	49.01	**0.8532**	0.0452	0.2947	0.1716
	LP-BNN + GN (ours)	47.15	**0.8553**	**0.0577**	0.2866	0.1623

Tab. 2: Comparative results obtained on the OOD task for semantic segmentation. The results are averaged over three seeds.

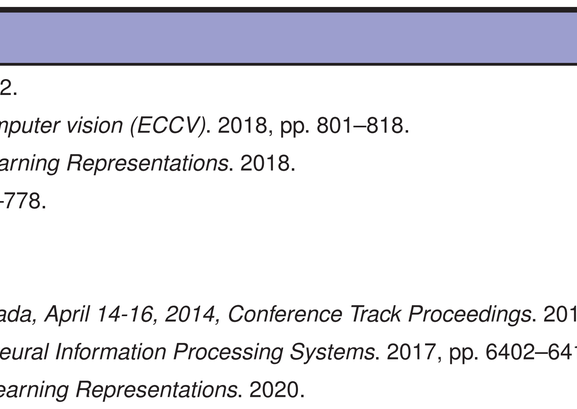

Fig. 4: Visual assessment on two images of BDD-Anomaly in which a motorcycle (OOD class) is present. For each image: on the first row - input image and confidence maps for MCP, BE and LP-BNN (ours); on the second row - GT segmentation and segmentation maps for MCP, BE and LP-BNN (ours). LP-BNN is less confident on the OOD objects.

Conclusions

We propose a new BNN framework able to quantify uncertainty in the context of deep learning. Owing to each layer of the network being tied to and regularized by a VAE, LP-BNNs are stable, efficient, and therefore easy to train compared to existing BNN models. The extensive empirical comparisons on multiple tasks show that LP-BNNs reach state-of-the-art levels with substantially lower computational cost. We hope that our work will open new research paths on effective training of BNNs. In the future we intend to explore new strategies for plugging more sophisticated VAEs in our models along with more in-depth theoretical studies.

References

[Blu+15] Charles Blundell et al. "Weight Uncertainty in Neural Networks". In: *International Conference on International Conference on Machine Learning*. 2015, pp. 1613–1622.

[Che+18] Liang-Chieh Chen et al. "Encoder-decoder with atrous separable convolution for semantic image segmentation". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 801–818.

[HD18] Dan Hendrycks and Thomas Dietterich. "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations". In: *International Conference on Learning Representations*. 2018.

[He+16] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[Hen+19] Dan Hendrycks et al. "A Benchmark for Anomaly Segmentation". In: *arXiv preprint arXiv:1911.11132* (2019).

[KH+09] Alex Krizhevsky, Geoffrey Hinton, et al. *Learning multiple layers of features from tiny images*. Tech. rep. Citeseer, 2009.

[KW14] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. 2014.

[LPB17] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. "Simple and scalable predictive uncertainty estimation using deep ensembles". In: *Advances in Neural Information Processing Systems*. 2017, pp. 6402–6413.

[WTB20] Yeming Wen, Dustin Tran, and Jimmy Ba. "BatchEnsemble: an Alternative Approach to Efficient Ensemble and Lifelong Learning". In: *International Conference on Learning Representations*. 2020.