Monte Carlo Dropout Based BatchEnsemble For Improving Uncertainty Estimation

Shubham Jain*
Indian Institute of Technology, Hyderabad, India
Sangareddy, Telangana, India
shubham.jain.jb@gmail.com

P.K. Srijith †
Indian Institute of Technology, Hyderabad, India
Sangareddy, Telangana, India
srijith@cse.iith.ac.in

ABSTRACT

Modelling uncertainty in deep learning is important for several high-risk applications such as autonomous driving and healthcare. Existing techniques for uncertainty modelling in deep learning such as Monte Carlo (MC) Dropout [1] and BatchEnsemble [2] suffer from some drawbacks. MC dropout shares parameters across models resulting in highly correlated predictions while BatchEnsemble requires storing additional parameters for each model in the ensemble. In our work, we aim to bring the best of both worlds by combining MC-dropout in the process of ensemble creation in BatchEnsemble. The proposed approach, Monte-Carlo BatchEnsemble, helps in generating ensembles with less correlation in prediction with the addition of a few parameters. The experimental results show the effectiveness of the proposed technique for image classification.

ACM Reference Format:
Shubham Jain and P.K. Srijith . 2023. Monte Carlo Dropout Based BatchEnsemble For Improving Uncertainty Estimation. In *6th Joint International Conference on Data Science & Management of Data (10th ACM IKDD CODS and 28th COMAD) (CODS-COMAD 2023), January 4–7, 2023, Mumbai, India.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3570991.3571038

1 MONTE CARLO BATCHENSEMBLE

Deep learning models have brought a lot of advances in several domains of AI, however they are not effective in modelling uncertainty in predictions. Bayesian deep learning models based on BatchEnsemble and Monte Carlo (MC) dropout provide a practical approach to model uncertainty for many real-world applications. MC dropout creates an ensemble of neural networks by giving multiple passes enabling dropout at inference time, however this model uses the same set of parameters leading to highly-correlated predictions. BatchEnsemble overcomes this by creating neural networks with different sets of weights, but consequently requires additional parameters to be stored for each model in the ensemble. We propose Monte Carlo BatchEnsemble that combines the ideas of MC Dropout and BatchEnsemble to develop more efficient deep ensembles with a reduced number of parameters.

Following BatchEnsemble, we maintain shared weights $W \in R^{m \times n}$ in every layer across all ensembles. To create ensembles, we multiply the shared weight W with many sets of two one-ranked vectors $\alpha_i \in R^m$ and $\beta_i \in R^n$ called fast learning weights. Unlike ensembles, we don't maintain separate α_i and β_i for each model in the ensemble. Some of the models share the fast learning weights but use MC dropout on these fast learning weights while forming the ensemble. This allows us to form multiple models with the same fast learning weights. In MC BatchEnsemble, we consider an element-wise product of β_i vector with $v \in [0, 1]^n$ where elements of v vector are sampled from Bernoulli distribution with drop probability of p. This results in new vector $\beta'_i = \beta_i \odot v$. We get multiple sets of weights $\bar{W}_i$ given by $\bar{W}_i = W \circ Z_i$, where $Z_i = \alpha_i \beta_i'^{\top}$. If we consider K sets of α and β to create the ensemble, MC BatchEnsemble can effectively create of $K \times L$ ensembles, with L dropouts on the fast learning weight β. If we keep high number of K (with multiple sets of α and β) and a low number of L (forward passes with dropout), it will lead to less time for inference by parallelizing the forward pass but results in a higher number of parameters. Consequently, we can trade-off between the time of inference, number of parameters and Ensemble correlation.

2 RESULTS

We performed image classification using a wide-resnet-28x10(Sergey et. al. (2017)[3]). We used CIFAR10 for training and evaluated it on SVHN and CIFAR100, which act as out-of-distribution data for our model. In our work, we used 8 sets of 2 learnable, one-ranked vectors to create 8 ensembles. Also, we gave our model 8 forward passes by dropping some of the values of the β vector. This setup allowed us to create $8 \times 8 = 64$ ensembles but only requires storing 8 different α and β vectors. We calculate predictive entropy to measure the uncertainty. Our model (MC_BE) got a predictive entropy of 2.293 and 2.309 compared to 0.522 and 0.497 in MC dropout and 2.274 and 2.283 in BatchEnsemble on CIFAR100 and SVHN respectively as OOD data. A comparison on other metrics is in Table 1.

Table 1: Comparisons of approaches trained on CIFAR10

Metric	MC Dropout	BE	MC_BE
Test Accuracy	94.25	94.32	93.22
Parameters	36489290	38205168	36695912
Inference Time	0.05 sec	0.02 sec	0.03 sec

CODS-COMAD 2023, January 4–7, 2023, Mumbai, India

ACM ISBN 978-1-4503-9797-1/23/01.
https://doi.org/10.1145/3570991.3571038

REFERENCES

[1] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, Vol. 48. PMLR, 1050–1059.
[2] Yeming Wen and Dustin Tran. 2020. BatchEnsemble: An Alternative Approach to Efficient Ensemble and Lifelong Learning. *ICLR* abs/2002.06715 (2020).
[3] Sergey Zagoruyko and Nikos Komodakis. 2017. Wide Residual Networks.