Efficient Monte-Carlo Dropout for Uncertainty Quantification

Srinath Srinivasan
Dept. of Computer Science
North Carolina State University
Raleigh, USA
ssrini27@ncsu.edu

***Abstract*—Uncertainty quantification is important to realize the confidence of a model's prediction. There are several studied techniques for quantifying the uncertainty of machine learning models one of which is Monte-Carlo dropout. Due to the computational cost of performing inference time modifications to a trained model, Monte Carlo dropouts have rarely been used in performance-oriented machine learning applications. This article discusses a model-splitting technique to modify the Monte-Carlo dropout process to perform uncertainty quantification at the millisecond level. Preliminary analysis shows a potential 25-33 times speedup in performing Monte-Carlo dropouts making it usable for rapid inference.**

I. Introduction

Uncertainty quantification(UQ) is an important technique that can be used to understand the confidence of a model in its predictions. Once we have the confidence of the model, we can decide on whether to keep or discard the model's prediction based on the problem's requirements. There have been many methods proposed to quantify the uncertainty of Machine Learning(ML) models [1].

A traditional approach to quantify uncertainty would be to consider all weights of a neural network as random variables instead of a point value such as in Bayesian neural networks(BNN) [2]. Although these kinds of networks provide a confidence estimate in their predictions, their power cannot be realized due to the immense computation required to perform backpropagation on the Bayesian layers. Several approximations have been suggested out of which Monte-Carlo(MC) dropouts have shown promise due to their ease of implementation and their applicability to pre-trained networks.

In this study, we focus on accelerating the Monte-Carlo dropout process in a pre-trained model to provide model inference at the millisecond level.

II. Related work

Dropout [3] was introduced as a technique to avoid neural network overfitting. This technique is usually applied during the training process, however, Gal et al. [4] mathematically showed that dropouts during inference time (Monte-Carlo dropouts) could serve as an approximation to Bayesian methods such as in BNNs.

Sun et al. [5] proposed an efficient way to apply Monte-Carlo dropouts in spiking neural networks by taking advantage of their time-step mechanism. They proclaim that uncertainty estimation through Monte Carlo dropout enables the application of neural networks in high-stakes biological applications.

Huang et al. [6] propose a region-based temporal aggregation technique that exploits the temporal nature of the data to simulate the Monte-Carlo sampling process producing a 10x speedup.

Srinivasan et al. [7] studied uncertainty quantification in book genre classification by using confidence intervals to control the number of predictions that are filtered by the model in a multi-label classification task using bidirectional LSTM's. [8]

Singhal et al. [9] studied the feasibility of using MC dropout to filter out of distribution(OOD) samples while testing. They show that up to 80% of ignored samples are saved from misclassification when classifying noisy images.

III. Methodology

The first part of this section explains how UQ is done on a Natural Language Processing(NLP) task using MC-dropout. The second part explains how model-splitting is applied to the existing technique to achieve inference speedup.

A. Modeling uncertainty in a NLP task

Sentiment analysis is a well-studied task [10] in machine learning. In a NLP context, sentiment analysis involves analyzing text to understand the sentiment of the author. A news headlines dataset [11] is chosen and the task is to classify whether a headline is sarcastic or not.

From the introduction of transformers [12], they have been widely used in NLP tasks [13]. Transformer-based models have shown state-of-the-art results in sentiment analysis. We choose to build off of a pre-trained RoBERTa model [14] trained on the Twitter emotion dataset. To this model, an additional dropout and fully connected layer are added and fine-tuned with our dataset.

The added dropout layer is enabled with inference time dropouts to generate slightly different outputs during inference time. Inputs are passed into the network multiple times during inference. Therefore we can quantify our model's uncertainty by calculating statistics on the outputs of the model.

B. Accelerating the UQ process

From our model design, we can see that a major overhead while performing multiple inferences is calculating the forward pass over the pre-trained RoBERTa model multiple times. We refer to the model before our first MC-dropout enabled layer as the model body and the rest of the model as the model head. The outputs from the model body for a given input remain constant over the multiple iterations that we need to perform our UQ technique.

It is only necessary to calculate the forward pass of the body of the model once and pass its outputs to the model head over the multiple iterations. This technique of splitting the model head from the model body is referred to as model-splitting. The model splitting process is illustrated in Fig.1. Intuitively, it is easy to realize why model-splitting can save on compute time while performing multiple forward passes for UQ during inference.

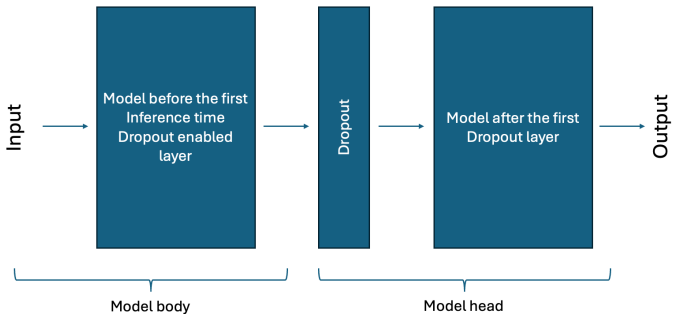

Fig. 1. Model splitting technique illustrated

IV. Experiments and Results

This section describes the experimental setup used, the experiments performed, and the results of those experiments.

A. Experimental setup

1) Dataset - News headlines: The news headlines dataset contains 28,503 headlines and their appropriate labels denoting whether the headline is sarcastic or not making our task a binary classification problem. The dataset is very clean and usable without any extra data-cleaning process. For our problem, we choose to ignore the website link for each news headline that is included in the dataset.

The dataset is split into an 80:10:10 ratio for training, validation, and testing respectively. The data is tokenized using the WordPiece [15] tokenization process.

2) Model: A pre-trained RoBERTa model trained on the TweetEval dataset is chosen as the model body and one dropout and fully-connected layer are added on top as the model head. The dropout rate is kept at 0.2 during both training and inference. The model body's weights are kept frozen and only the head is trained for 5 epochs with a starting learning rate of 0.0001 and using the AdamW [16] optimizer. A linear learning rate scheduler is also used and a batch size of 32 is used during training. The outputs of the network are passed through a Sigmoid activation to force them between 0 and 1. The model architecture is as described in Fig. 2.

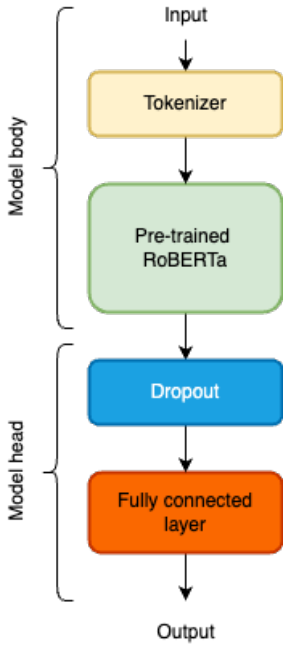

Fig. 2. Diagrammatic view of the model architecture used for experiments

B. Uncertainty Quantification

First, we describe the MC dropout process for uncertainty quantification without model-splitting.

To perform UQ using MC-dropout, we enable dropout during inference time on the dropout layer added at the model head. It is to be noted that by definition, the model head begins at the layer from which inference time dropouts are enabled. During inference, we perform the forward pass over multiple times to get slightly different outputs from the model due to dropout changing the weight matrix randomly. The process is visualized in Fig.3.

Now with the outputs from multiple iterations i, we calculate the mean μ and standard deviation σ. With this, we define a confidence interval based on these metrics. The frequentist confidence interval is given by (1). Here Z is the z-table value of the percentage confidence interval.

$$(\mu - Z\sigma, \mu + Z\sigma) \tag{1}$$

It is to be noted that here we are assuming that the outputs are in the form of a Gaussian, however, the technique of calculating uncertainty is not dependent on the distribution of the network outputs. The interval can simply be viewed as a way to include or exclude predictions based on how far they are from a set threshold value.

Once we have the confidence interval we decide whether to keep the prediction or discard it based on where it lies with respect to the set threshold value T. As the network's outputs lie between 0 and 1, we select our threshold as 0.5

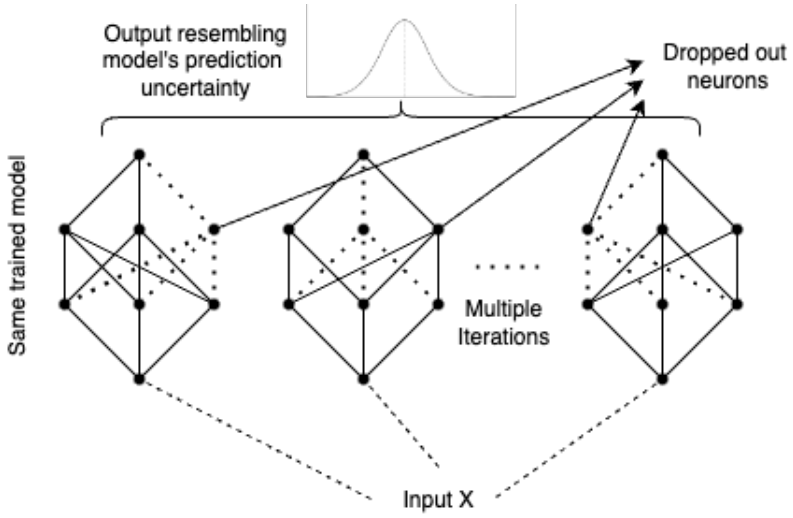

Fig. 3. Monte-Carlo dropout process for uncertainty quantification

and keep predictions whose confidence intervals lie completely on either side of the threshold value. If the confidence interval contains the threshold value, we choose to ignore the network's prediction.

C. Experiments

1) Baseline model performance: We first benchmark the model's performance on the test dataset containing 2851 news headlines. We test the model with and without Monte Carlo dropout to understand how dropouts during inference time affect the network's outputs and the results are as in Table I.

TABLE I
F1 SCORES OF MODEL WITH AND WITHOUT MC-DROPOUT ENABLED ON THE TEST SET

Model type	F1 score
Original model	0.92
MC-dropout enabled model	0.88

We observe a drop of 0.04 in the F1 score calculated from the MC-dropout enabled model, this is expected due to the loss of information due to dropout.

2) Using UQ to extract good quality predictions: Next, we perform UQ using the MC-dropout technique to extract good-quality predictions while rejecting the bad ones. Table II describes the results of calculating uncertainty over many iterations.

TABLE II
RESULTS OF APPLYING UNCERTAINTY QUANTIFICATION WITH STATISTICS CALCULATED OVER MULTIPLE ITERATIONS

Statistics calculated over	F1 Score	No. of samples ignored
50 iterations	0.94	50
100 iterations	0.94	55

We observe that the UQ technique allows us to improve the F1 score by 0.02 while ignoring around 50 samples. We do not see a significant improvement by increasing the number of iterations over which statistics are calculated. It is also interesting to observe how the F1 score improves by considering multiple outputs of the MC-dropout-enabled model in comparison to a single prediction from the MC-dropout-enabled model. This increase of 0.06 in the F1 score highlights the ensembling effect of considering multiple slightly different outputs from the same model.

It is also important to consider how many of these ignored predictions were "saved" from being misclassified. We find that around 70% of the samples were misclassified by the original model. This means we "saved" around 35 test samples from being misclassified.

3) Measuring inference time: We compare the inference time of making multiple forward passes for UQ during inference time in Table III. We observe a significant improvement in the time taken to make multiple predictions using model-splitting over using the whole MC-dropout enabled model. We also see a slight increase in the time taken to make a single prediction with model-splitting. This increase is due to manually routing the outputs of the model body to the model head.

TABLE III
TIME TO PERFORM MULTIPLE FORWARD PASSES DURING INFERENCE TIME USING THE WHOLE MODEL AND SPLIT MODEL IN MS

Model type	1 iteration	50 iterations	100 iterations
UQ-model	1.91ms	95.8ms	190ms
UQ-model with model splitting	1.94ms	3.78ms	5.61ms

To better understand the performance improvement with the model splitting technique we can describe the improvement as a 25x speedup in the case of 50 iterations and a 33x speedup

in the case of performing 100 iterations. We plot the same result in Fig.4.

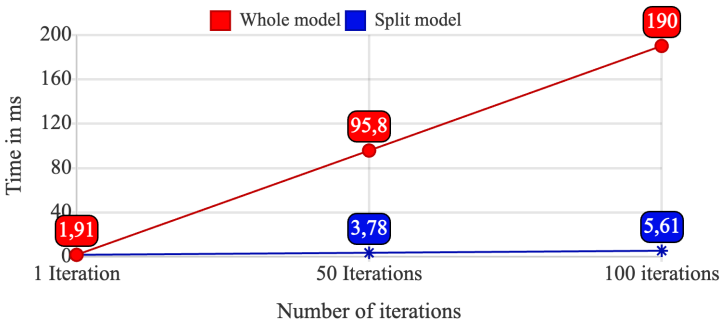

Fig. 4. Inference time improvement by using model splitting

V. Conclusion

Uncertainty quantification remains one of the bigger challenges in the adoption of machine learning models across sensitive, mission-critical sectors. Monte-Carlo dropouts show great potential in quantifying the uncertainty of models for various use cases. Accelerating the process of uncertainty quantification through Monte Carlo dropouts could increase the adoption of the technique to fields with millisecond-level inference requirements. The model-splitting technique discussed in this work shows a 25-33x speedup for quantifying the uncertainty of a model trained for a sentiment analysis task. The model-splitting technique could allow many previous works to be used in practice that may have been considered infeasible due to inference costs.

VI. Future work

Current effort measures performance of model-splitting on a simple problem with the model head consisting of a single dropout and fully connected layer. Further, increasing model head complexity could potentially increase inference costs. It is important to understand the benefit of Monte Carlo dropouts with model-splitting in complex machine-learning problems.

Acknowledgment

Special thanks to Dr. Dongkuan Xu for his guidance and advice that shaped this work as a part of the course project in CSC-591: Efficient Deep Learning at NC State University.

References

[1] V. Nemani, L. Biggio, X. Huan, Z. Hu, O. Fink, A. Tran, Y. Wang, X. Zhang, and C. Hu, "Uncertainty quantification in machine learning for engineering design and health prognostics: A tutorial," *Mechanical Systems and Signal Processing*, vol. 205, p. 110796, 2023.

[2] R. Neal, "Bayesian learning via stochastic dynamics," in *Advances in Neural Information Processing Systems*, S. Hanson, J. Cowan, and C. Giles, Eds., vol. 5. Morgan-Kaufmann, 1992.

[3] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[4] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*. PMLR, 2016, pp. 1050–1059.

[5] T. Sun, B. Yin, and S. Bohté, "Efficient uncertainty estimation in spiking neural networks via mc-dropout," in *International Conference on Artificial Neural Networks*. Springer, 2023, pp. 393–406.

[6] P.-Y. Huang, W.-T. Hsu, C.-Y. Chiu, T.-F. Wu, and M. Sun, "Efficient uncertainty estimation for semantic segmentation in videos," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 520–535.

[7] S. Srinivasan, S. G. Shivanirudh, S. Sathya, and T. T. Mirnalinee, "Exploring bayesian uncertainty modeling for book genre classification," in *2022 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, 2022, pp. 179–183.

[8] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm networks," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 4, 2005, pp. 2047–2052 vol. 4.

[9] R. Singhal and S. Srinivasan, "Extracting usable predictions from quantized networks through uncertainty quantification for ood detection," 2024.

[10] M. Wankhade, A. C. S. Rao, and C. Kulkarni, "A survey on sentiment analysis methods, applications, and challenges," *Artificial Intelligence Review*, vol. 55, no. 7, pp. 5731–5780, Oct 2022. [Online]. Available: https://doi.org/10.1007/s10462-022-10144-1

[11] R. Misra and P. Arora, "Sarcasm detection using news headlines dataset," *AI Open*, vol. 4, pp. 13–18, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666651023000013

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023.

[13] N. Patwardhan, S. Marrone, and C. Sansone, "Transformers in the real world: A survey on nlp applications," *Information*, vol. 14, no. 4, 2023. [Online]. Available: https://www.mdpi.com/2078-2489/14/4/242

[14] F. Barbieri, J. Camacho-Collados, L. Espinosa Anke, and L. Neves, "TweetEval: Unified benchmark and comparative evaluation for tweet classification," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, T. Cohn, Y. He, and Y. Liu, Eds. Online: Association for Computational Linguistics, Nov. 2020, pp. 1644–1650. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.148

[15] M. Schuster and K. Nakajima, "Japanese and korean voice search," *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5149–5152, 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:22320655

[16] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.