Quantization of Bayesian Neural Networks and Its Effect on Quality of Uncertainty

Mahesh Subedar [*1] **Ranganath Krishnan** [*1] **Sidharth N. Kashyap** [2] **Omesh Tickoo** [1]

Abstract

We present a Post-Training Quantization (PTQ) flow for Bayesian Neural Networks (BNNs) to reduce the memory and compute requirements. The mean-field variational inference with Gaussian distribution (mean μ and standard deviation σ) in BNNs have 2x the number of parameters and memory footprint compared to the deterministic Deep Neural Networks (DNNs) with similar architectures. During BNN inference runs, multiple (T) stochastic forward-passes through the network require T times more compute and bandwidth requirements. An 8-bit representation (INT8) for the sampled weights reduces the bandwidth requirements by 200x for 50 Monte Carlo stochastic forward-passes. We present an end-to-end BNN quantization flow with up to 7.1x reduction in model size has no significant degradation in the test accuracy or the quality of uncertainty as compared to the full-precision FP32 model. We also show that representing the σ values at lower precision (1-8 bits) gives comparable results to INT8 representation.

1. Introduction

Deep neural networks (DNNs) are unable to capture model uncertainty associated with their predictions, which is essential for reliable decision-making. Bayesian neural networks (BNNs) (MacKay, 1992; Neal, 1995) can capture principled uncertainty estimates including aleatoric and epistemic uncertainties (Kendall & Gal, 2017). BNNs are demonstrated to be robust (Ovadia et al., 2019; Krishnan & Tickoo, 2020) and widely used in many safety-critical applications including autonomous driving (McAllister et al., 2017; Michelmore et al., 2020), medical diagnosis (Filos et al., 2019; Kompa et al., 2021), video activity recognition (Subedar et al., 2019). The additional computation and memory cost associated with BNNs rise practical challenges for wider usage and deployment. Various hardware acceleration approaches for BNNs (Cai et al., 2018; Awano & Hashimoto, 2020; Fan et al., 2021) are being proposed more recently to address computation challenges. To realize optimal hardware acceleration, model quantization plays an important role. Neural network quantization reduces the model size and memory footprint by reducing the number of bits required to represent the model parameters and activations with the goal of improving inference throughput and energy cost. Quantization of Deep Neural Networks (DNNs) (Han et al., 2015; Dong et al., 2017; Banner et al., 2018) is a well-studied problem. Though quantization of DNNs is an active area of research and has shown promising results, the quantization of BNNs and the effect towards the quality of uncertainty is not yet studied well to the best of our knowledge.

In this work, we study the effect of post-training quantization of BNNs with variational layers modeled through mean-field Gaussian distributions parameterized by mean (μ) and standard deviation (σ). The quantization of μ and σ values is critical to reduce the memory footprint while representing the activations and sampled weights in quantized 8-bit format will reduce the compute and bandwidth requirements. Compared to DNN with similar neural network architecture, BNN has twice the number of parameters and requires multiple stochastic forward passes while sampling the weights from the posterior distribution. Since BNNs have higher compute and memory requirements, they can significantly benefit from the quantization.

Contributions: Our main contributions in this work are:

- proposed a post-training quantization flow for BNNs and evaluated the effect on model accuracy and quality of uncertainty,
- presented a systematic study on the effect of BNN quantization under dataset shifts,
- evaluated various bit representations for the variational parameters representing the weight posterior in BNNs demonstrating there is no significant degradation in model accuracy and quality of uncertainty.

[*]Equal contribution [1]Intel Labs, Hillsboro, Oregon (USA) [2]Intel Corporation. Edinburgh, Scotland (UK). Correspondence to: Mahesh Subedar <mahesh.subedar@intel.com>.

Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning.,

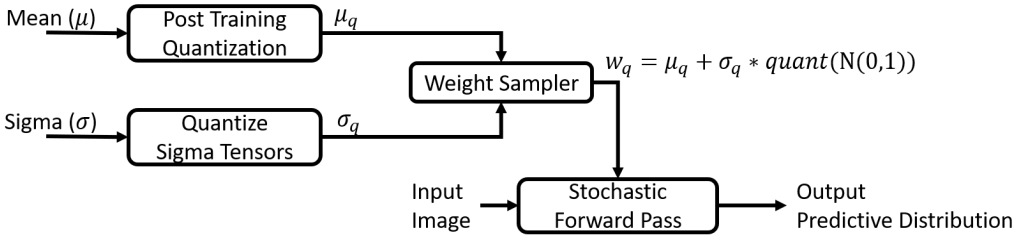

Figure 1. Post-training quantization steps for Bayesian Neural Networks (BNNs): Mean (μ) and standard deviation (σ) are the variational parameters representing Gaussian distribution, and μ_q, σ_q are their quantized representations, respectively. The sampled weights w_q use the same calibration step as the μ_q values. The stochastic forward-passes use INT8 computations.

2. Background

A low-precision representation of neural network parameters and activations is beneficial in real-world deployments. A combined pruning, quantization, and Huffman coding (Han et al., 2015) steps are shown to provide a significant reduction in memory and compute requirements for DNNs. The changes in architecture definition (Iandola et al., 2016) to reduce the number of parameters can further reduce the complexity. Binarized neural networks (Hubara et al., 2016) are the extreme case of representing the weights by binary values. Architectural changes including channel shuffle operation (Zhang et al., 2018) and depth-wise convolution are shown to provide efficient hardware implementation. Joint optimization of balancing depth, width and resolution for convolutional layers (Tan & Le, 2019) can provide improved accuracies at the reduced number of parameters. Please refer to the additional references (Sze et al., 2017; Zmora et al., 2019) for a more complete review of the efficient implementation of DNNs. Additional methods such as weight pruning and compression (Han et al., 2015) applied to DNNs help to further reduce the resource requirements. These added steps are orthogonal to the quantization step and applicable to BNNs, which can further improve the efficiencies of BNN inference, but we are not studying them in this paper.

The weights in BNNs are modeled as probability distributions. Given observed data $D\ \{\mathrm{x}, \mathrm{y}\}$, prior distribution $p(\mathrm{w})$ and model likelihood $p(\mathrm{y} \mid \mathrm{x}, \mathrm{w})$, posterior distribution over the weights $p(\mathrm{w}|\mathrm{D})$ is to be inferred. Prediction of an input is obtained through Bayesian marginalization with Monte Carlo sampling over the weight posterior. It is difficult to compute exact posterior in neural networks as it is analytically intractable, hence various approximate methods are proposed including variational inference (Graves, 2011; Blundell et al., 2015; Kingma et al., 2015) and stochastic-gradient MCMC (Welling & Teh, 2011; Zhang et al., 2020). In this work, we focus on mean-field variational inference where weights are modeled with fully factorized Gaussian distributions parameterized by variational parameters μ and σ i.e. $\mathrm{w} \sim \mathcal{N}(\mathrm{w} \mid \mu, \sigma)$. We refer to Blundell et al. (2015) for details on variational inference in BNNs.

Algorithm 1 quantization of σ

Input: σ (per layer tensor array)
nbits (bit-precision or number of bits)
Output: σ_q (per layer quantized tensor array)

$\sigma_{max} = max(\sigma)$
$\sigma_{min} = min(\sigma)$
if $\sigma_{max} == \sigma_{min}$ **then**
  $\sigma_{scale} = \sigma_{max}$
  $\sigma_{zero\_point} = 0$
else
  $min\_val = 0$
  $max\_val = 2^{nbits} - 1$
  compute $\sigma_{scale} = \frac{(max\_val - min\_val)}{(\sigma_{max} - \sigma_{min})}$
  compute $\sigma_{zero\_point} = min\_val - int(\frac{\sigma_{min}}{\sigma_{scale}})$
end if
compute $\sigma_q = \sigma_{zero\_point} + \sigma / \sigma_{scale}$

In this work, we evaluate the effect of the quantization of variational parameters in BNN on the quality of uncertainty using calibration metrics. Bhatt et al. (2020) summarize various calibration metrics as model calibration acts as a form of quality assurance for uncertainty estimates. We use Expected Calibration Error (ECE) (Naeini et al., 2015), Uncertainty Calibration Error (UCE) (Laves et al., 2019) and proper scoring rules (Gneiting & Raftery, 2007) such as Negative Log-Likelihood (NLL) and Brier score (Brier, 1950) calibration metrics to study the effect of quantization.

3. Implementation Details

Figure 2 shows the proposed PTQ flow for BNNs. The first step is to obtain a PTQ model for the mean (μ) values from the learned Gaussian distribution. We use the PyTorch PTQ steps (PyTorch-Documentation) to obtain a quantized INT8 model. The μ represents mid-point of the distribution

Table 1. Test accuracy (Test Acc), Expected Calibration Error (ECE), Negative Log-likelihood (NLL) and Brier scores for the BNN ResNet20 and BNN LeNet-5 models evaluated on CIFAR-10 and MNIST respectively using proposed PTQ flow.

DATASET	FORMAT	MODEL SIZE	TEST ACC (%) ↑	ECE (%)↓	NLL ↓	1+BRIER ↓
CIFAR10	FP32	4.37 MB	91.00 ± 0.07	1.523 ± 0.13	0.263 ± 0.0008	0.130 ± 0.0003
	INT8	1.09 MB	90.87 ± 0.15	2.001 ± 0.18	0.269 ± 0.0015	0.132 ± 0.0006
	INT8_SIGMA4	0.87 MB	90.92 ± 0.05	1.778 ± 0.21	0.266 ± 0.0027	0.131 ± 0.0007
	INT8_SIGMA2	0.72 MB	90.85 ± 0.04	2.547 ± 0.01	0.273 ± 0.0018	0.134 ± 0.0005
	INT8_SIGMA1	0.54 MB	90.96 ± 0.10	0.711 ± 0.11	0.266 ± 0.0023	0.130 ± 0.0005
MNIST	FP32	0.51 MB	99.40 ± 0.01	0.221 ± 0.012	0.020 ± 0.0002	0.01 ± 0.0000
	INT8	0.13 MB	99.36 ± 0.02	0.211 ± 0.025	0.020 ± 0.0003	0.01 ± 0.0001
	INT8_SIGMA4	0.10 MB	99.36 ± 0.01	0.215 ± 0.024	0.020 ± 0.0002	0.01 ± 0.0001
	INT8_SIGMA2	0.08 MB	99.32 ± 0.01	0.277 ± 0.033	0.024 ± 0.0003	0.01 ± 0.0000
	INT8_SIGMA1	0.06 MB	99.34 ± 0.01	0.351 ± 0.018	0.027 ± 0.0003	0.01 ± 0.0000

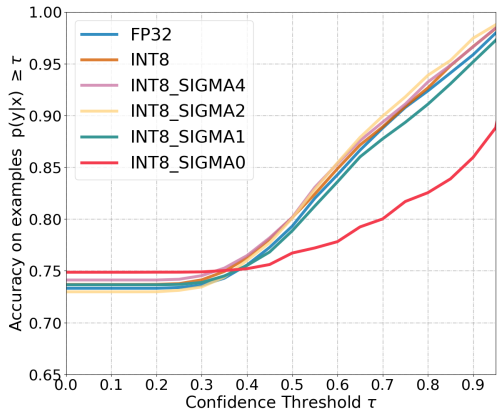

Figure 2. BNN ResNet-20/CIFAR10: Model confidence evaluation under dataset shift. Accuracy as a function of confidence evaluation on CIFAR-10 data shifted with Gaussian blur at intensity 3.

and hence will resemble the point estimates for the DNNs. We use the calibration step from μ to quantize w_q. The *fbgemm* (Khudia et al., 2021) backend at 8-bit precision is set to run the quantized models on x86 machines. We independently obtain per-channel quantized σ values represented using Uniform symmetric quantizer (Krishnamoorthi, 2018). Algorithm 1 provides the steps to calculate *scale* and *mid_point* values for quantized σ representation. The $quant$ function to calculate sigma values is given by Equation 1.

$$\sigma_q = \sigma_{zero\_point} + \sigma/\sigma_{scale} \tag{1}$$

For every Monte Carlo (MC) stochastic forward-pass, we calculate the sampled weights using reparameterization equation (Blundell et al., 2015):

$$w_q = \mu_q + \sigma_q * quant(N(0,1)) \tag{2}$$

where, μ_q, σ_q are the quantized mean and sigma values for the Gaussian distributions, respectively; N is the sample from INT8 quantized Gaussian (zero mean, unit variance) distribution. Quantized INT8 weight values w_q use the calibration values obtained during PTQ of μ_q. The output of the stochastic forward passes provides posterior prediction distribution.

We evaluate our experiments with BNN ResNet-20 (He et al., 2015) and LeNet-5 (LeCun et al., 2015) architectures on CIFAR-10 (Krizhevsky et al., 2009) and MNIST datasets respectively. We train the BNN ResNet-20 model on the CIFAR-10 dataset in full-precision floating-point (FP32) format using Adam optimizer for 200 epochs with initial learning rate of 0.001 and batch size of 128. The initial learning rate was multiplied by 0.1, 0.01, 0.001 and 0.0005 at epochs 80, 120, 160 and 180, respectively as part of the learning rate scheduler. The BNN LeNet-5 model was trained with MNIST dataset in full-precision floating-point (FP32) format using Adam optimizer for 80 epochs with an initial learning rate of 0.01 and batch size of 128. The initial learning rate was multiplied by 0.1 and 0.01 at epochs 40 and 60 respectively as part of the learning rate scheduler. During the inference step, we run 50 MC stochastic forward passes using the quantized neural network weights sampled from the learned posterior Gaussian distribution. The resulting predictive distribution is utilized to compute test accuracy, and model calibration metrics to evaluate the quality of uncertainty.

In addition to INT8 representation for σ_q, we studied the 4, 2 and 1 quantized representation (henceforth referred as INT8_SIGMAn, where n=$\{1,2,4\}$ indicates bits used to represent σ) resulting in model size reduction by 5.3x, 6.4x and 7.1x respectively as compared to FP32 format.

4. Results

We evaluate the effect of quantization on the quality of uncertainty in BNNs under dataset shift. We utilize 16 different types of image corruptions (Hendrycks & Dietterich, 2018)

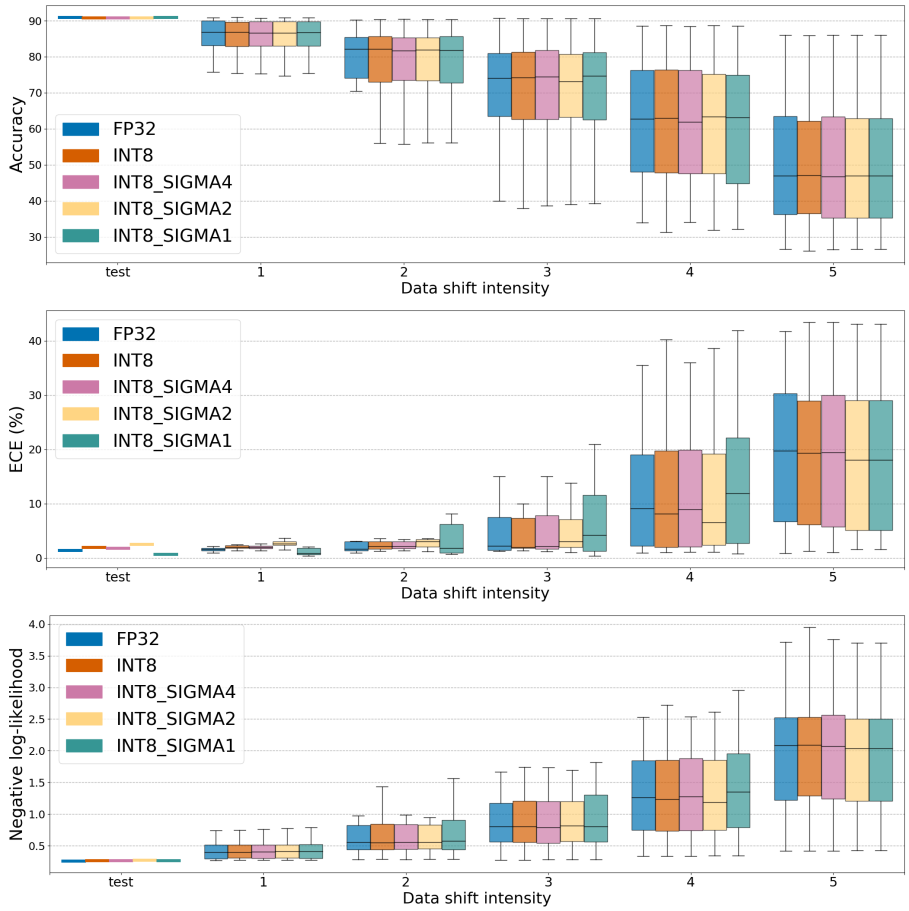

Figure 3. BNN ResNet-20/CIFAR-10: Evaluation of various bit-precision for quantized BNN model under dataset shift at various shift intensities (1-5). At each shift intensity level, the boxplot summarizes the results across 16 different dataset shift types showing the min, max and quartiles. We observe there is no significant degradation in test accuracy and model calibration with BNN quantization.

at 5 different levels of intensities for each datashift type on CIFAR-10. Table 1 shows the test accuracy and model calibration metrics for the quantized models compared to the full-precision FP32 model, indicating there is no significant degradation in the model performance while representing BNN model parameters with lower bit-precisions. We compare the test accuracy, Expected Calibration Error (ECE) and Negative Log-likelihood (NLL) metrics under dataset shift (shown in Figure 3) for full-precision FP32 and quantized models (additional metrics including UCE and Brier score are provided in the Appendix). The box plot shows BNN models can be quantized effectively without sacrificing the accuracy and quality of uncertainty even under dataset shift. Figure 2 shows the model yield higher confidence when they are accurate for both full-precision and quantized models. The INT8, INT8_SIGMA4, INT8_SIGMA2 and INT8_SIGMA1 quantized models have lower memory footprint, representing the weights in quantized format reduce the bandwidth requirement by 4Tx, where T is the number of stochastic forward passes (T=50 in our experiments corresponds to 200x reduction in local bandwidth to caches).

5. Discussion and Conclusion

We presented an end-to-end PTQ flow in INT8 representation for BNN quantization. Our results indicate INT8 BNN model with up to 7.1x reduction in model size (with different bit-precisions for σ_q along with INT8 μ_q) has no degradation in the test accuracy or the calibration error compared to the end-to-end processing of FP32 model results.

Lower precision representation for σ values did not affect the quality of uncertainty scores. The analysis on the perturbed weights from 1-bit σ_q representation indicates the sampled parameters albeit quantized have a similar range of values as the higher precision σ_q representation. Hence, we do not see any degradation in test accuracy or the model calibration error. These systematic studies demonstrate the applicability of quantized BNN models in resource-constrained applications with limited compute budget.

References

Awano, H. and Hashimoto, M. Bynqnet: bayesian neural network with quadratic activations for sampling-free uncertainty estimation on fpga. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1402–1407. IEEE, 2020.

Banner, R., Hubara, I., Hoffer, E., and Soudry, D. Scalable methods for 8-bit training of neural networks. *arXiv preprint arXiv:1805.11046*, 2018.

Bhatt, U., Antorán, J., Zhang, Y., Liao, Q. V., Sattigeri, P., Fogliato, R., Melançon, G. G., Krishnan, R., Stanley, J., Tickoo, O., et al. Uncertainty as a form of transparency: Measuring, communicating, and using uncertainty. *Proceedings of the AAAI/ACM Conference on Artificial Intelligence, Ethics, and Society*, 2020.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pp. 1613–1622. PMLR, 2015.

Brier, G. W. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

Cai, R., Ren, A., Liu, N., Ding, C., Wang, L., Qian, X., Pedram, M., and Wang, Y. Vibnn: Hardware acceleration of bayesian neural networks. *ACM SIGPLAN Notices*, 53 (2):476–488, 2018.

Dong, X., Chen, S., and Pan, S. J. Learning to prune deep neural networks via layer-wise optimal brain surgeon. *arXiv preprint arXiv:1705.07565*, 2017.

Fan, H., Ferianc, M., Rodrigues, M., Zhou, H., Niu, X., and Luk, W. High-performance fpga-based accelerator for bayesian neural networks. *arXiv preprint arXiv:2105.09163*, 2021.

Filos, A., Farquhar, S., Gomez, A. N., Rudner, T. G., Kenton, Z., Smith, L., Alizadeh, M., de Kroon, A., and Gal, Y. A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks. *arXiv preprint arXiv:1912.10481*, 2019.

Gneiting, T. and Raftery, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Graves, A. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pp. 2348–2356, 2011.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition, 2015.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2018.

Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., and Bengio, Y. Binarized neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 4114–4122, 2016.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

Kendall, A. and Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pp. 5574–5584, 2017.

Khudia, D., Huang, J., Basu, P., Deng, S., Liu, H., Park, J., and Smelyanskiy, M. Fbgemm: Enabling high-performance low-precision deep learning inference. *arXiv preprint arXiv:2101.05615*, 2021.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. *arXiv preprint arXiv:1506.02557*, 2015.

Kompa, B., Snoek, J., and Beam, A. L. Second opinion needed: communicating uncertainty in medical machine learning. *NPJ Digital Medicine*, 4(1):1–6, 2021.

Krishnamoorthi, R. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.

Krishnan, R. and Tickoo, O. Improving model calibration with accuracy versus uncertainty optimization. *Advances in Neural Information Processing Systems*, 33, 2020.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Laves, M.-H., Ihler, S., Kortmann, K.-P., and Ortmaier, T. Well-calibrated model uncertainty with temperature scaling for dropout variational inference. *arXiv preprint arXiv:1909.13550*, 2019.

LeCun, Y. et al. Lenet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*, 20(5):14, 2015.

MacKay, D. J. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

McAllister, R., Gal, Y., Kendall, A., Van Der Wilk, M., Shah, A., Cipolla, R., and Weller, A. Concrete problems for autonomous vehicle safety: advantages of bayesian deep learning. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 4745–4753, 2017.

Michelmore, R., Wicker, M., Laurenti, L., Cardelli, L., Gal, Y., and Kwiatkowska, M. Uncertainty quantification with statistical guarantees in end-to-end autonomous driving control. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7344–7350. IEEE, 2020.

Naeini, M. P., Cooper, G., and Hauskrecht, M. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Neal, R. M. Bayesian learning for neural networks. *PhD thesis, University of Toronto*, 1995.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

PyTorch-Documentation. Post training quantization in PyTorch. URL https://pytorch.org/docs/stable/quantization.html.

Subedar, M., Krishnan, R., Meyer, P. L., Tickoo, O., and Huang, J. Uncertainty-aware audiovisual activity recognition using deep bayesian variational inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Sze, V., Chen, Y.-H., Yang, T.-J., and Emer, J. S. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329, 2017.

Tan, M. and Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6105–6114. PMLR, 2019.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688, 2011.

Zhang, R., Li, C., Zhang, J., Chen, C., and Wilson, A. G. Cyclical stochastic gradient mcmc for bayesian deep learning. *International Conference on Learning Representations*, 2020.

Zhang, X., Zhou, X., Lin, M., and Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6848–6856, 2018.

Zmora, N., Jacob, G., Zlotnik, L., Elharar, B., and Novik, G. Neural network distiller: A python package for dnn compression research. *arXiv preprint arXiv:1910.12232*, 2019.

6. Appendix

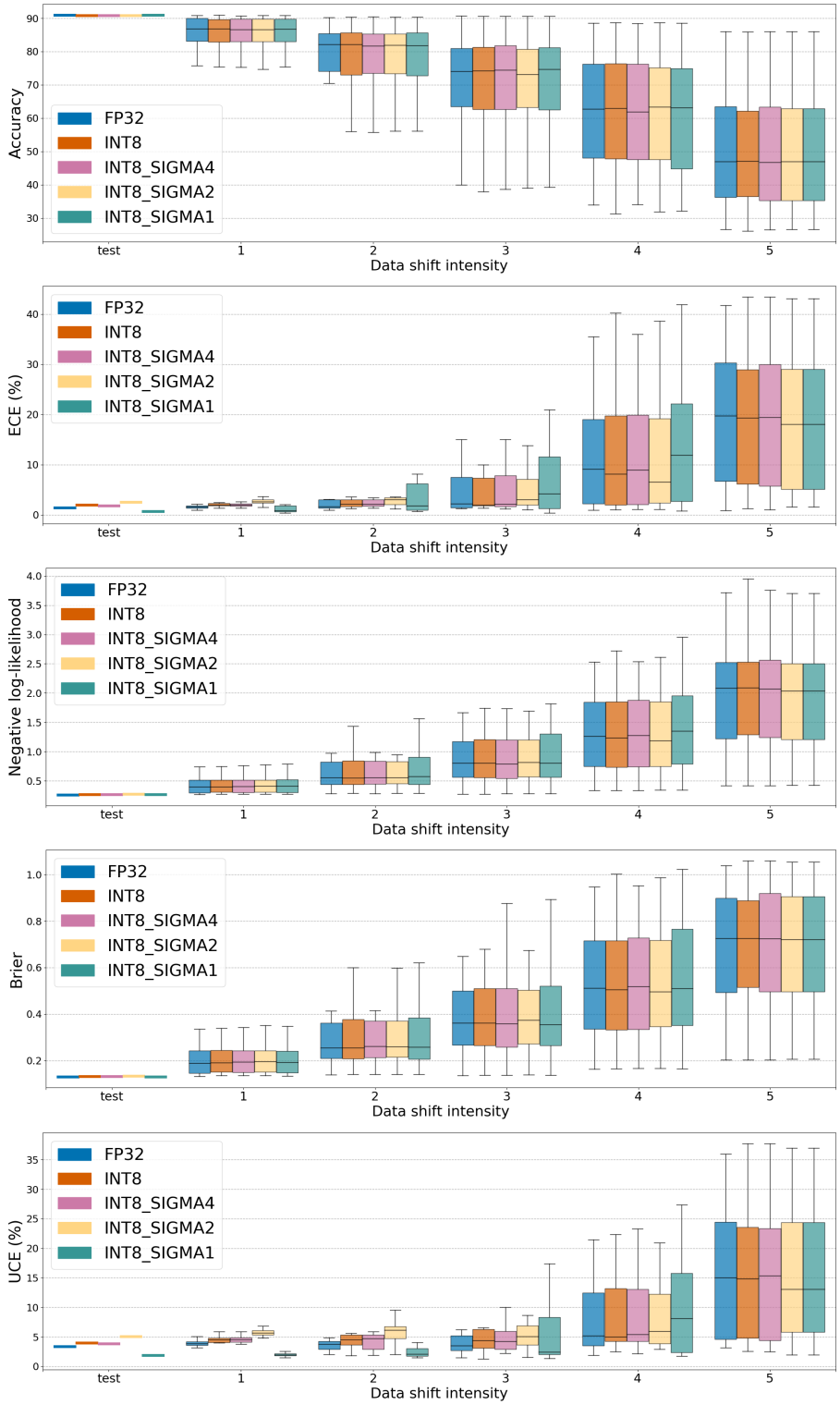

Figure 4. BNN ResNet-20/CIFAR-10: Evaluation of various bit-precision for quantized BNN model under dataset shift at various shift intensities (1-5). At each shift intensity level, the boxplot summarizes the results across 16 different datashift types showing the min, max and quartiles. We observe there is no sigificant degradation in test accuracy and model calibration with BNN quantization.

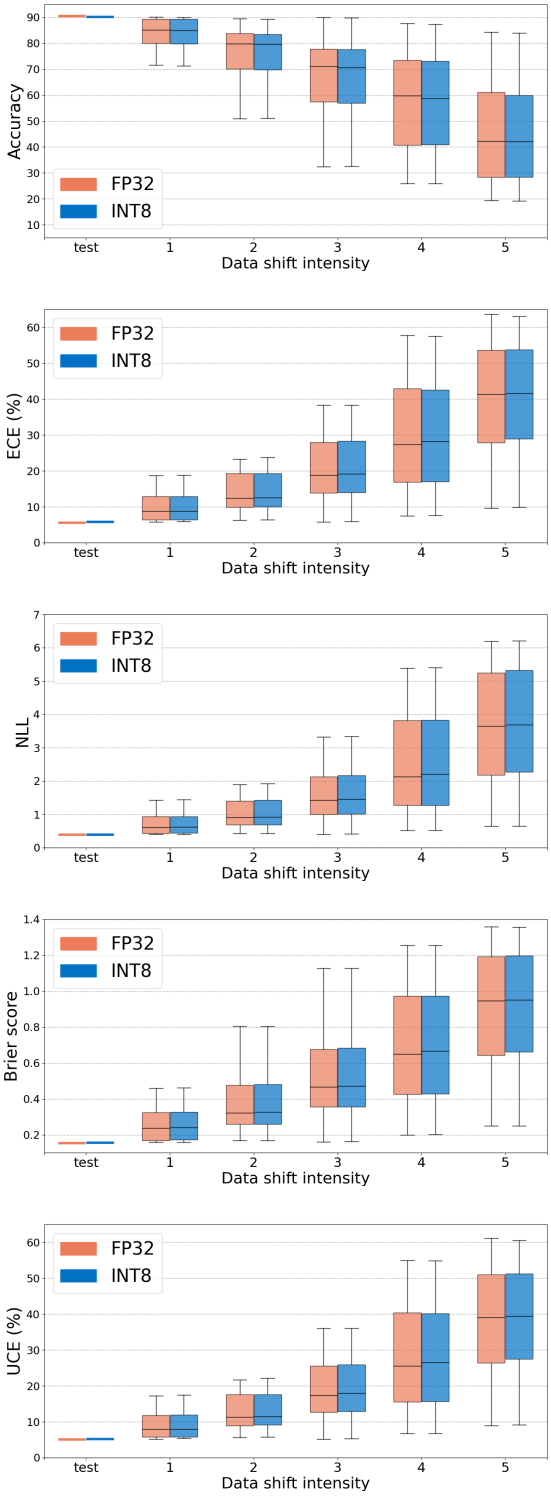

Figure 5. DNN ResNet-20/CIFAR-10: Evaluation of various bit-precision for quantized DNN model under dataset shift at various shift intensities (1-5). At each shift intensity level, the boxplot summarizes the results across 16 different datashift types showing the min, max and quartiles. We observe there is no sigificant degradation in test accuracy and model calibration with DNN quantization.