On the Effects of Quantisation on Model Uncertainty in Bayesian Neural Networks (Supplementary material)

Martin Ferianc[1] **Partha Maji**[2] **Matthew Mattina**[3] **Miguel Rodrigues**[1]

[1]Electronic and Electrical Engineering Department, University College London, London, UK
[2]Arm ML Research Lab, Cambridge, UK
[3]Arm ML Research Lab, Boston, USA

A DETAIL OF BAYESIAN NEURAL NETWORK METHODS

We will illustrate the complications of the different approaches for Bayesian inference with the help of a toy example: a common linear layer with an input matrix $\boldsymbol{f}_i \in \mathbb{R}^{I\times M}$ with I samples and M features and some weights $\boldsymbol{f}_w \in \mathbb{R}^{M\times F}$ with the output $\boldsymbol{f}_o \in \mathbb{R}^{I\times F}$ with F output features such that $\boldsymbol{f}_o = \boldsymbol{f}_i\boldsymbol{f}_w$.

Monte Carlo Dropout The concept of MCD (Gal and Ghahramani, 2015) lays in casting dropout (Srivastava et al., 2014) training in NNs as approximate Bayesian inference. Dropout can be described by applying a random element-wise mask $\boldsymbol{K} \in \mathbb{R}^{I\times M}; \boldsymbol{K} \sim \text{Bernoulli}(p)$ of zeros and ones with probability $0 \leq p \leq 1$ to the input $\boldsymbol{f}_i$ and scaling the non-zero elements by $\frac{1}{1-p}$ as $\boldsymbol{f}_o = \boldsymbol{f}_w\left(\frac{1}{1-p}\boldsymbol{K}\odot\boldsymbol{f}_i\right)$. The authors of MCD show that the use of dropout in NNs before every weight-bearing layer can be interpreted as a Bayesian approximation and by applying dropout, it can approximate the integral over the models' weights (Gal and Ghahramani, 2015). Therefore, to estimate the predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{x}^*)$ it is needed to collect the results of L forward passes, while sampling and applying the element-wise masks. Training is usually done through a single sample. Therefore, the only implementation-wise complications of this method are the need for random generation of zeros and ones and their subsequent element-wise application. The number of parameters, and thus memory footprint, stays constant.

Bayes-By-Backprop In BBB (Blundell et al., 2015) the weight uncertainty is modelled explicitly, by assuming an approximation $q(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta})$ for the posterior $p(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{y})$ with respect to learnable parameters $\boldsymbol{\theta}$. The learning is performed through minimising the distance bound between $q(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta})$ and $p(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{y})$. The most common approximation q for weights $\boldsymbol{w}$ is a mean-field approximation, such that $\boldsymbol{w}{\sim}\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\sigma}^2)$, with individual mean $\boldsymbol{\mu}\in\mathbb{R}^{M\times F}$ and variance $\boldsymbol{\sigma}^2 \in \mathbb{R}^{M\times F}$ for each weight where $\boldsymbol{\theta} = \{\boldsymbol{\mu},\boldsymbol{\sigma}^2\}$ and $\mathcal{N}$ represents a Gaussian distribution (Kingma and Welling, 2013; Ranganath et al., 2014). Kingma and Welling (2013) have introduced the *reparametrisation trick*, that allows sampling of the weights with respect to the q, such that $\boldsymbol{f}_w = \boldsymbol{\mu} + \boldsymbol{\epsilon}\odot\phi(\boldsymbol{\sigma})$ where $\boldsymbol{\epsilon}{\sim}\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$, $\boldsymbol{I}$ is an identity and $\phi(.)$ is a positive-forcing function e.g. softplus. The sampled $\boldsymbol{w}$ can then be used, such that $\boldsymbol{f}_o = \boldsymbol{f}_i\boldsymbol{f}_w$. Similarly to MCD, to estimate the predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{x}^*)$ it is needed to collect the results of L forward passes with respect to L weight samples. Training is usually done through a single sample. This method requires the ability of the hardware to efficiently sample a more complex, Gaussian distribution. Moreover, this model uses double the number of parameters for the same network size, due to the means paired with variances.

Stochastic Gradient Langevin Dynamics with Hamiltonian Monte Carlo In comparison to the previous two approaches, in SGHMC (Chen et al., 2014), it is not necessary to perform sampling and random number generation during evaluation. The $\boldsymbol{w}_l$ corresponding to a single set of weights from the ensemble can then be used directly during evaluation instead of sampling them via $\boldsymbol{w}_l{\sim}p(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{y})$ as in the case of the two previous methods. To obtain $p(\boldsymbol{y}^*|\boldsymbol{x}^*)$ it is needed to collect the results of L forward passes with respect to the ensemble with L members corresponding to L weights $\boldsymbol{w}$. Training is performed similarly to standard pointwise NNs. In comparison to the previous two methods, this method does not require sampling of a distribution during evaluation. However, it requires $L\times$ more memory resources in comparison to pointwise NNs, to store the entire ensemble. At the same time, it is necessary to consider the extra time needed to load the weights $\boldsymbol{w}$ to memory.

B METRICS

In addition to measuring the root-mean-squared error (RMSE) and the classification error, we establish metrics for the evaluation of the quantified uncertainty.

B.1 NEGATIVE-LOG LIKELIHOOD

Based on (Lakshminarayanan et al., 2017), our base metric for evaluating the quality of the predictive uncertainty is the negative-log likelihood (NLL). We chose averages in our evaluations, due to easier interpretability and consistence. In the regression case, NLL can be formulated with respect to a single-valued Gaussian as in equation (1).

$$\text{NLL} = \frac{1}{N}\sum_{n=1}^{N} \frac{\log \sigma_w^2(\boldsymbol{x}_n)}{2} + \frac{(y_n - \mu_w(\boldsymbol{x_n}))^2}{2\sigma_w^2(\boldsymbol{x}_n)} + \log\sqrt{2\pi}. \quad (1)$$

In the classificaiton case, NLL can be formulated with respect to cross-entropy as in equation (2).

$$\text{NLL} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} y_n^k \log \mu_w^k(\boldsymbol{x_n}) \quad (2)$$

B.2 PREDICTIVE ENTROPY

In case of classification for which the labels are not available, which is the case for most out-of-distribution datasets, we measure the quality of the uncertainty prediction with respect to the average predictive entropy (aPE) as in equation (3).

$$\text{aPE} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} \mu_w^k(\boldsymbol{x_n}) \log \mu_w^k(\boldsymbol{x_n}) \quad (3)$$

B.3 EXPECTED CALIBRATION ERROR

Additionally, we measure the calibration of the BNNs and their sensitivity through expected calibration error (ECE) (Guo et al., 2017). ECE relates confidence with which a network makes predictions to accuracy, thus it measures whether a network is over-confident or under-confident in its predictions, with respect to the softmax output. To compute ECE, the authors propose to discretize the prediction probability interval into a fixed number of bins, and assign each probability to the bin that encompasses it. The ECE is the difference between the fraction of predictions in the bin that are correct (accuracy) and the mean of the probabilities in the bin (confidence). ECE computes a weighted average of this error across bins as shown in equation (4), where n_b is the number of predictions in bin b and accuracy(b) and confidence(b) are the accuracy and confidence of bin b, respectively. We set $B = 10$.

$$\text{ECE} = \sum_{b=1}^{B} \frac{n_B}{N} |\text{accuracy}(b) - \text{confidence}(b)| \quad (4)$$

To summarise, for regression we are measuring RMSE and NLL and for image classification problems we were observing the classification error, NLL, aPE and ECE.

C ADDITIONAL RESULTS

In this Section we report additional measurements with respect to test data, confusion data or the domain shifts for the image classification problems.

C.1 MNIST

The additional results for MNIST-based experiments are presented in Figures 1 (a-e), 2 (a-e) and 3 (a,b). Starting with the results for changing the activation precision, it can be seen that as the predictive entropy for the confusion data increases, the entropy for the test data stays constant and it increases near the smallest bit-width, as seen in Figure 1 (a). As a result, the ECE, in Figure 1 (b), for the test data also increases and that is due to 2 reasons: *1)* The quantisation collapses when $n = 3$; *2)* the predictive uncertainty increases disproportionately to the error, which can be observed in SGHMC and BBB. Subsequently, the NLL on the test data stays constant and increases in the collapsed regime as seen in Figure 1 (c). Nevertheless, as supported by the ECE plot for the confusion data in Figure 1 (d), by becoming more uncertain, the activation quantisation reduced the confidence of BNNs on the confusion dataset, which is desired. This can be also observed on the NLL for the confusion data in Figure 1 (e). Comparing these results to Figures 2 (a-e), it can be seen that weight quantisation follows similar trends, and the ECE and the predictive entropy collapse, when n reaches extrema. Looking in more detail at the aPE (a) and NLL (b) on test data with augmentations in Figures 3 when $n = 7$ for activations and $n = 8$ for weights, it can be seen that the methods remain robust under augmentations, but at the same time they are more uncertain. This further supports the claim that the BNNs can indeed be already quantised to approximately 8-bit integer representation. In addition, we present 32-bit floating-point results with respect to Figures 4 (a-d) and there are barely any visible deviations with respect to the quantised counterparts.

C.2 CIFAR-10

The additional results for CIFAR-10-based experiments are presented in Figures 5 (a-e), 6 (a-e) and 7 (a,b). Similarly to MNIST experiments, as seen in Figures 5 (a-e), it can be observed that as the accuracy degrades, so does rightfully the predictive confidence, best seen in Figure 5 (a). At the same time the ECE on the test data also increases, and on the confusion data it tends to decrease, as seen in Figures 5 (b,d). However, this time it is due to the collapse in the activations, where the bit-width is not satisfactory and a more advanced quantisation scheme might be needed for $n \leq 4$. The results for weight quantisation support these observations as seen in Figures 6 (a-e), nevertheless with smaller impact on the ranges of the deviations. This suggests a potential investigation into mixed precision representation with low bit-width of weights, while preserving the activation precision. As seen in Figure 7 (a,b) presenting results on test data with augmentations when $n = 7$ for activations and $n = 8$ for weights, it can be seen that the quantisation does not prevent the Bayesian inference methods from quantifying uncertainty in their predictions. In addition, we present 32-bit floating-point results with respect to Figures 8 (a-d) and there are barely any visible deviations with respect to the quantised counterparts presented in the supplementary material or the main body of the paper.

REFERENCES

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*.

Chen, T., Fox, E., and Guestrin, C. (2014). Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pages 1683–1691.

Gal, Y. and Ghahramani, Z. (2015). Dropout as a bayesian approximation. *arXiv preprint arXiv:1506.02157*.

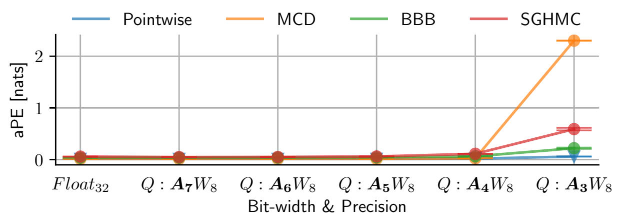
Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. *arXiv preprint arXiv:1706.04599*.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

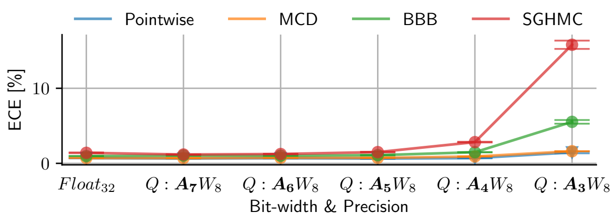
Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413.

Ranganath, R., Gerrish, S., and Blei, D. (2014). Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822.

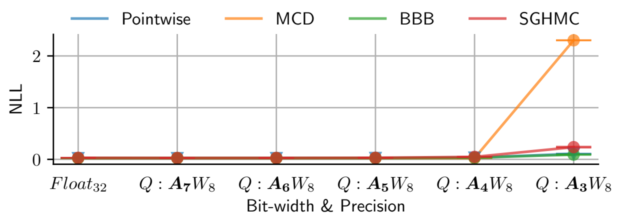
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.



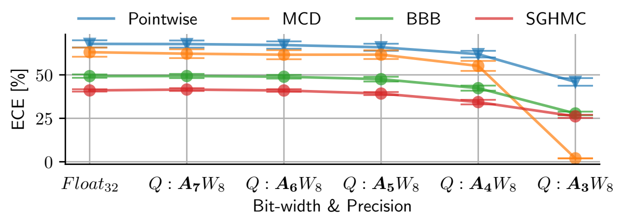
(a)



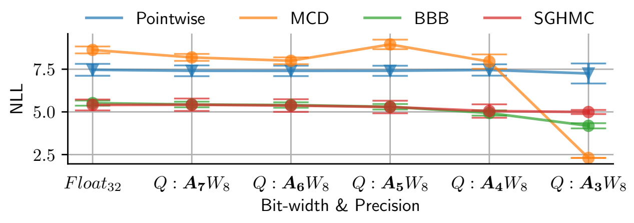
(b)



(c)

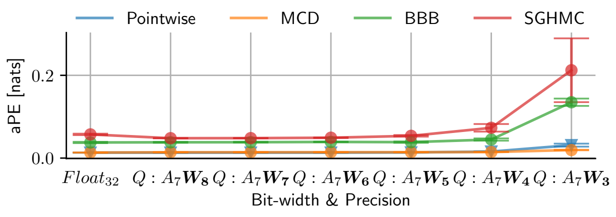


(d)

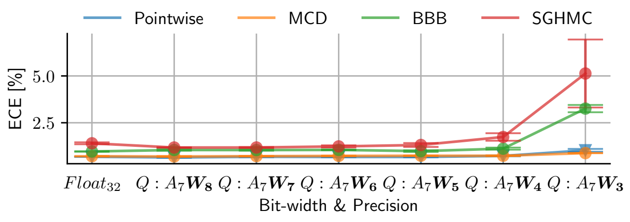


(e)

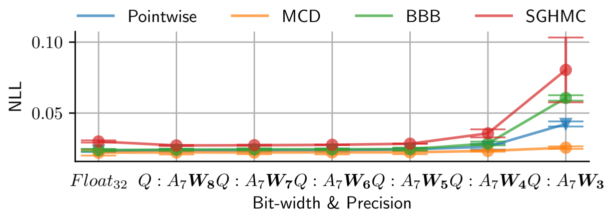
Figure 1: *Changing activation precision, fixing weight precision.* MNIST results with respect to average predictive entropy (aPE) (a), expected calibration error (ECE) (b) and negative log-likelihood (NLL) (c) on test data and ECE (d) and NLL (e) on FashionMNIST. Q stands for quantised activations (A) and weights (W). Subscript denotes bit-width. MCD collapses when the bit-width ≤ 3 for A.



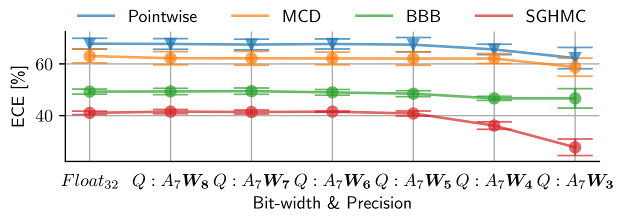
(a)



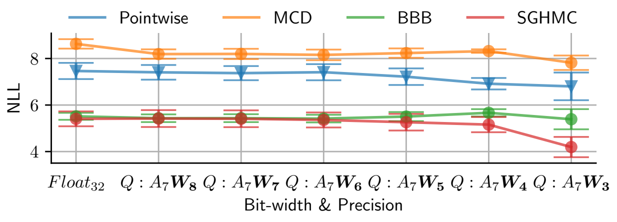
(b)



(c)

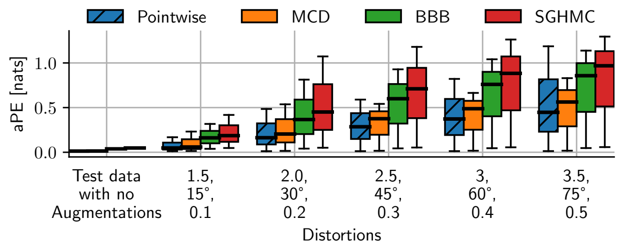


(d)

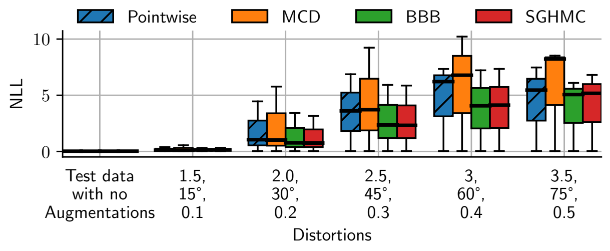


(e)

Figure 2: *Fixing activation precision, changing weight precision.* MNIST results with respect to average predictive entropy (aPE) (a), expected calibration error (ECE) (b) and negative log-likelihood (NLL) (c) on test data and ECE (d) and NLL (e) on FashionMNIST data. Q stands for quantised activations (A) and weights (W). Subscript denotes bit-width.

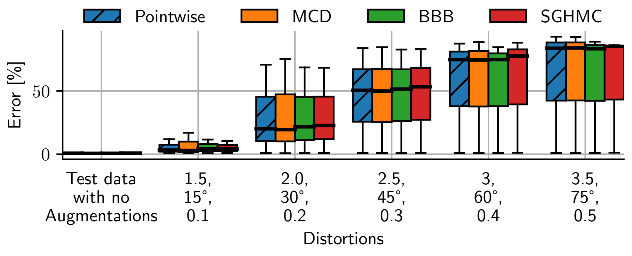


(a)

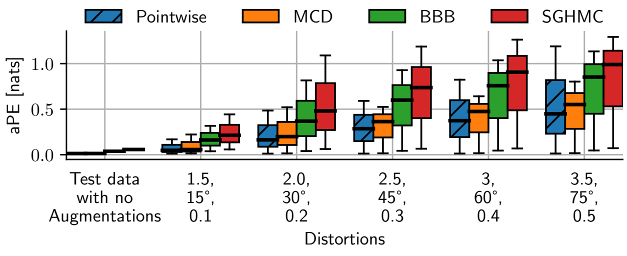


(b)

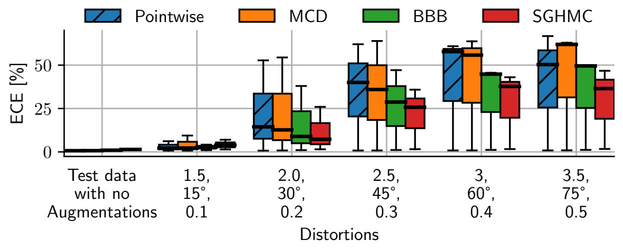
Figure 3: Average predictive entropy (aPE) (a) and negative log-likelihood (NLL) (b) with respect to 7-bit activations and 8-bit weights and three augmentations applied to the MNIST test set: Brightness [1.5-3.5], Rotation [15°-75°] and Horizontal shift [0.1-0.5 of image size].



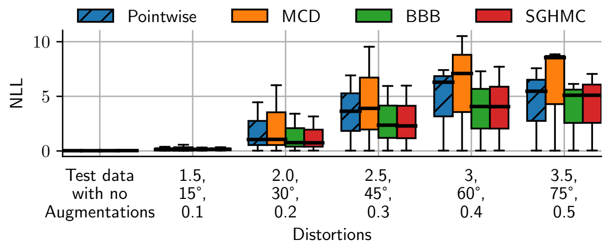
(a)



(b)

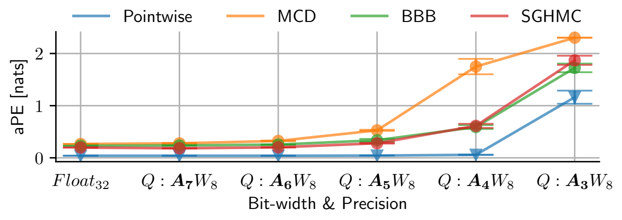


(c)

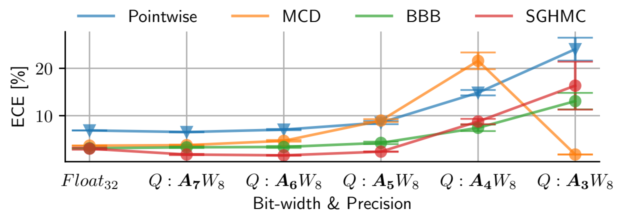


(d)

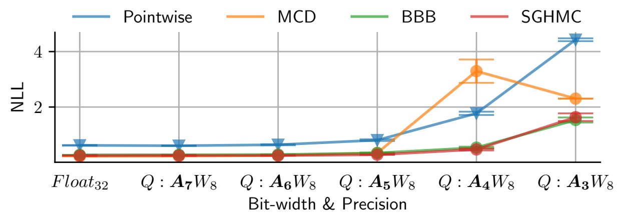
Figure 4: Classification error (a), average predictive entropy (aPE) (b), expected calibration error (ECE) (c) and negative log-likelihood (NLL) (d) with respect to 32-bit floating-point activations and weights and three augmentations applied to the MNIST test set: Brightness [1.5-3.5], Rotation [15°-75°] and Horizontal shift [0.1-0.5 of image size].



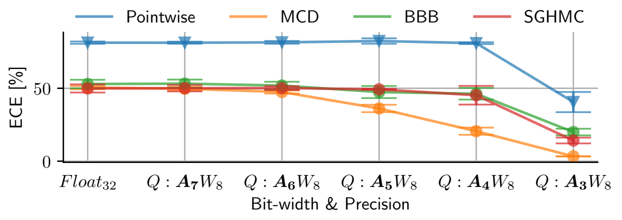
(a)



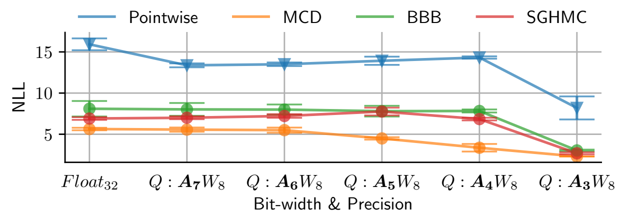
(b)



(c)

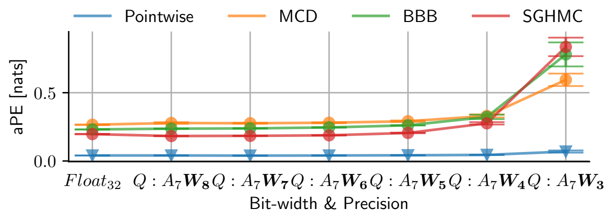


(d)

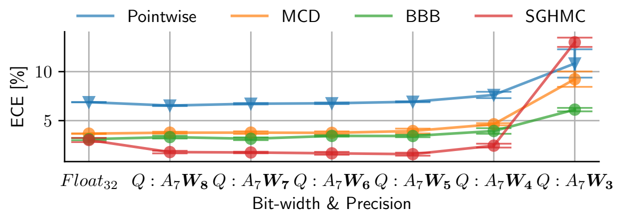


(e)

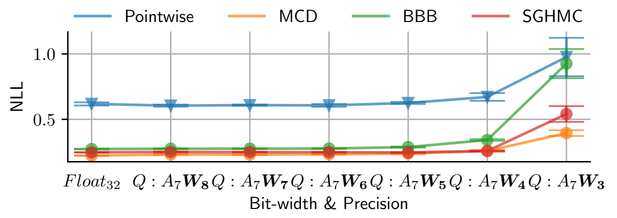
Figure 5: *Changing activation precision, fixing weight precision.* CIFAR-10 results with respect to average predictive entropy (aPE) (a), expected calibration error (ECE) (b) and negative log-likelihood (NLL) (c) on test data and ECE (d) and NLL (e) on SVHN. Q stands for quantised activations (A) and weights (W). Subscript denotes bit-width. All methods collapse when the bit-width ≤ 4 for A.



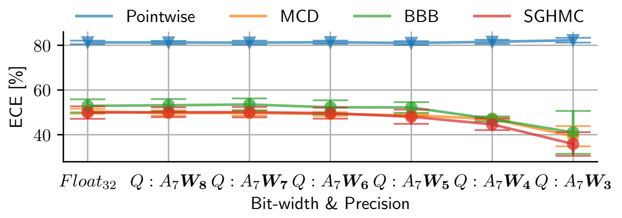
(a)



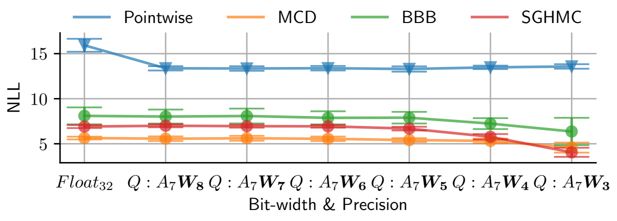
(b)



(c)

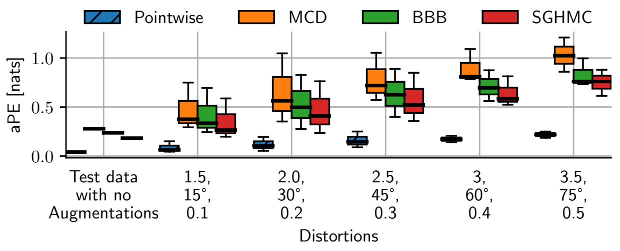


(d)

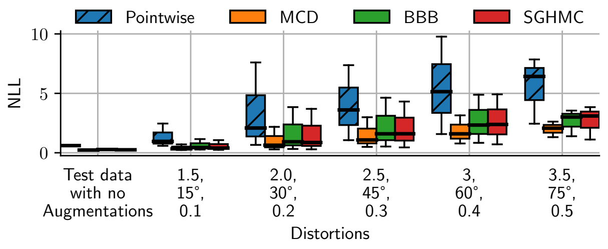


(e)

Figure 6: *Fixing activation precision, changing weight precision.* CIFAR-10 results with respect to average predictive entropy (aPE) (a), expected calibration error (ECE) (b) and negative log-likelihood (NLL) (c) on test data and ECE (d) and NLL (e) on SVHN. Q stands for quantised activations (A) and weights (W). Subscript denotes bit-width. All methods except MCD collapse when the bit-width ≤ 3 for W.

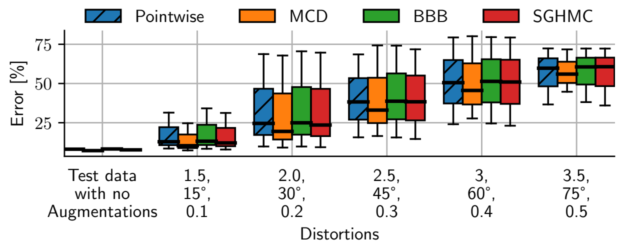


(a)

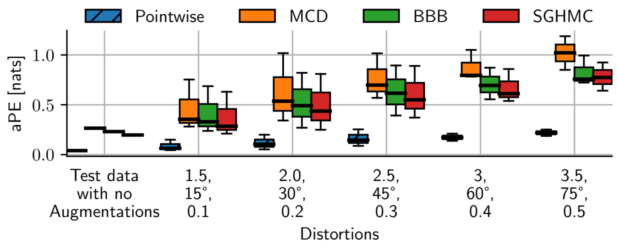


(b)

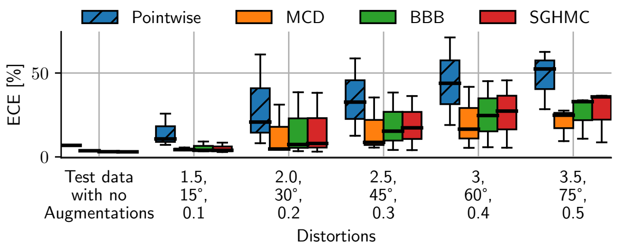
Figure 7: Average predictive entropy (aPE) (a) and negative log-likelihood (NLL) (b) with respect to 7-bit activations and 8-bit weights and three augmentations applied to the CIFAR-10 test set: Brightness [1.5-3.5], Rotation [15°-75°] and Horizontal shift [0.1-0.5 of image size].



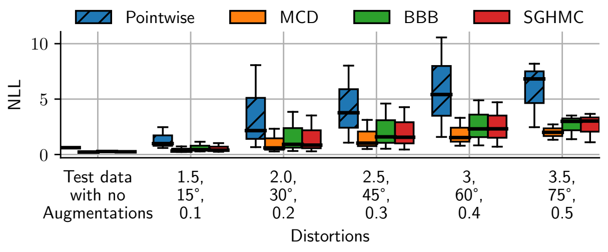
(a)



(b)



(c)



(d)

Figure 8: Classification error (a), average predictive entropy (aPE) (b) and expected calibration error (ECE) (c) and negative log-likelihood (NLL) (d) with respect to 32-bit floating-point activations and weights and three augmentations applied to the CIFAR-10 test set: Brightness [1.5-3.5], Rotation [15°-75°] and Horizontal shift [0.1-0.5 of image size].