On the Impact of Weight Quantization on Deep Neural Network Uncertainty

Shuang Liang[1,2], Xun Lu[2], Zi-Ang Liu[3], Ming-Liang Wang[3], Yan Lyu[3*], Shao-Qun Zhang[1,2*]

[1]National Key Laboratory for Novel Software Technology, Nanjing University, China
[2]School of Intelligent Science and Technology, Nanjing University, China
[3]China Mobile Zijin Innovation Institute, China
lvyansgs@js.chinamobile.com, zhangsq@lamda.nju.edu.cn

Abstract

Weight quantization (WQ) is a key technique for lightweight Deep Neural Network (DNN) computations. While existing algorithms often pursue memory compression and inference acceleration with accuracy comparable to full-precision models, the effect of WQ on DNN uncertainty remains largely unexplored. In this paper, we quantify the impact of WQ on DNN uncertainty through the novel Exact Moment Propagation (EMP) uncertainty estimator. It is observed that WQ significantly increases DNN uncertainty. Based on the EMP estimator, we propose the MOMent Alignment (MOMA) to reduce WQ-induced uncertainty and preserve the accuracy of weight-quantized DNNs. Empirical results across various DNN architectures and datasets validate the effectiveness of both EMP and MOMA methods.

1 Introduction

In recent years, Weight Quantization (WQ) has attracted increasing interest in lightweight computations of Deep Neural Networks (DNNs), particularly in resource-constrained environments (Gope, Dasika, and Mattina 2019; Shen et al. 2024). Seminal studies show that WQ algorithms achieve at least 16x model compression (Li et al. 2016) and 3x training acceleration (Courbariaux, Bengio, and David 2015) while maintaining comparable accuracy to full-precision counterparts (Zhu et al. 2017; Li et al. 2022); however, the extent to which WQ techniques affect the uncertainty associated with specific outcomes remains largely unexplored.

This work focuses on quantifying the impact of WQ on DNN uncertainty. In the past decades, significant efforts have been made to quantify uncertainty in DNNs without WQ, such as Deep Ensembles (Lakshminarayanan, Pritzel, and Blundell 2017) and Moment Propagation (MP) (Wu et al. 2019). In addition, research has also examined uncertainty quantification in weight-quantized networks, such as BLRNet (Peters and Welling 2018) and BayesBiNN (Meng, Bachmann, and Khan 2020). However, a unified measure to quantify model uncertainty before and after WQ is lacking. Without this measure, one cannot fairly quantify the impact of WQ on DNN uncertainty.

[*]Corresponding authors.

In this paper, we propose the Exact Moment Propagation (EMP), a theoretical uncertainty quantification method that is independent of weight precision. Based on the EMP, we observe that quantized models exhibit greater uncertainty than their full-precision counterparts. Inspired by research on uncertainty-aware training for neural networks (Tabarisaadi et al. 2022; Einbinder et al. 2022), we propose the MOMent Alignment (MOMA), an optimization method compatible with various weight quantization training algorithms, to maintain accuracy and reduce uncertainty in quantized DNNs. This work fills the gap in quantifying and optimizing the impact of WQ on DNN uncertainty.

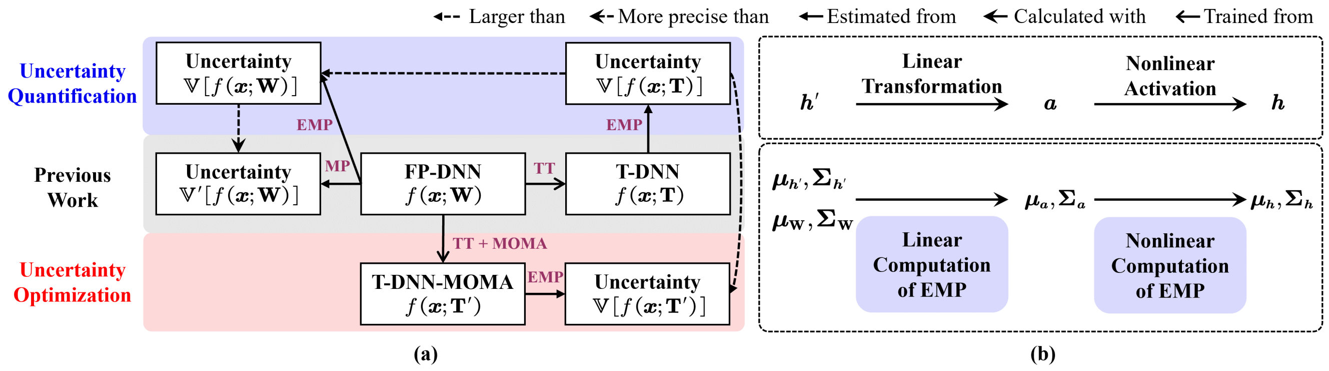
Our main contributions are illustrated in Figure 1(a) and further summarized as follows.

- We propose the EMP for DNN uncertainty quantification that is independent of weight precision in Algorithm 1. Empirical evidence in Subsection 5.1 shows that EMP achieves near-ideal uncertainty estimation performance, surpassing the MP, and that quantized DNNs exhibit greater uncertainty than their full-precision versions.
- We propose the MOMA for uncertainty optimization in Algorithm 2. Experiments in Subsection 5.2 show that MOMA greatly reduces the uncertainty of quantized models across different datasets.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 and Section 4 formally introduce the EMP and MOMA methods, respectively. Section 5 presents experiments on real-world datasets to validate the effectiveness of the EMP and MOMA methods. Section 6 concludes this work.

2 Related Work

Weight Quantization. WQ aims to encode model weights into a format of k bits, where $k = 1$ and $k = 1.58$ often lead to extreme compression. For $k = 1$, the weights of DNNs are compressed into binary values, such as $\{0, 1\}$ or $\{-1, 1\}$. Seminal works include BinaryConnect (Courbariaux, Bengio, and David 2015), Binarized Neural Networks (Hubara et al. 2016), and XNOR-Net (Rastegari et al. 2016). For $k = 1.58$, the weights are compressed into ternary values $\{-1, 0, 1\}$. Early milestone works include Ternary Weight Networks (Li et al. 2016), Trained



Larger than
More precise than
Estimated from
Calculated with
Trained from
Uncertainty Quantification
Previous Work
Uncertainty Optimization
Uncertainty $\mathbb{V}[f(\boldsymbol{x};\mathbf{W})]$
Uncertainty $\mathbb{V}[f(\boldsymbol{x};\mathbf{T})]$
Uncertainty $\mathbb{V}'[f(\boldsymbol{x};\mathbf{W})]$
FP-DNN $f(\boldsymbol{x};\mathbf{W})$
T-DNN $f(\boldsymbol{x};\mathbf{T})$
T-DNN-MOMA $f(\boldsymbol{x};\mathbf{T}')$
Uncertainty $\mathbb{V}[f(\boldsymbol{x};\mathbf{T}')]$
EMP
MP
TT
TT + MOMA
(a)
$\boldsymbol{h}'$
Linear Transformation
$\boldsymbol{a}$
Nonlinear Activation
$\boldsymbol{h}$
$\boldsymbol{\mu}_{\boldsymbol{h}'}, \boldsymbol{\Sigma}_{\boldsymbol{h}'}$
$\boldsymbol{\mu}_{\mathbf{W}}, \boldsymbol{\Sigma}_{\mathbf{W}}$
$\boldsymbol{\mu}_{\boldsymbol{a}}, \boldsymbol{\Sigma}_{\boldsymbol{a}}$
$\boldsymbol{\mu}_{\boldsymbol{h}}, \boldsymbol{\Sigma}_{\boldsymbol{h}}$
Linear Computation of EMP
Nonlinear Computation of EMP
(b)

Figure 1: (a) Overview of our work, where TT is short for Ternary Training, and (b) the workflow of the EMP.

Ternary Quantization (Zhu et al. 2017), and Ternary Connect (Lin et al. 2016). In recent years, ternary WQ has been successfully applied to transformer architectures (Vaswani et al. 2017), such as TernaryBERT (Zhang et al. 2020) and Q-ViT (Li et al. 2022). TernaryCLIP (Zhang et al. 2025) achieves extremely low-cost and high-efficiency computations in image-text contrastive modeling.

Uncertainty Quantification. To quantify the uncertainty in DNNs without WQ, Deep Ensembles (Lakshminarayanan, Pritzel, and Blundell 2017) trains independent networks in parallel, estimating predictive uncertainty via negative log likelihood. PL-DNN (Bibi, Alfadly, and Ghanem 2018) and MP (Wu et al. 2019) propagate input mean and covariance through the network to derive predictive uncertainty from output covariance. For uncertainty quantification in weight-quantized DNNs, BLRNet (Peters and Welling 2018) and BayesBiNN (Meng, Bachmann, and Khan 2020) adopt the Bayesian Neural Networks paradigm (Jospin et al. 2022), sampling the posterior distribution of weights to estimate predictive uncertainty through output covariance. Despite these advances, a unified uncertainty quantification method independent of weight precision is still lacking, precluding fair comparisons of DNN uncertainty before and after WQ.

3 Uncertainty Quantification

In this section, we propose the Exact Moment Propagation (EMP) method for quantifying the uncertainty of DNNs. In contrast to previous studies, which were designed solely for full-precision models and provided either exact output covariance under strong assumptions (Bibi, Alfadly, and Ghanem 2018) or approximated output covariance (Wu et al. 2019), the proposed EMP requires only mild assumptions, improves the precision of uncertainty estimation, and extends its applicability to quantized models. The EMP method consists of the linear and nonlinear computations, whose workflow is shown in Figure 1(b).

3.1 Problem Formulation

This subsection introduces the problem formulation of WQ and uncertainty quantification of DNNs.

Ternary Weight Quantization. Throughout this work, we focus on the ternary quantization and begin with an L-layer DNN $f(\boldsymbol{x};\mathbf{W})$, of which the forward propagation follows

$$\boldsymbol{h}^{(0)} = \boldsymbol{x}\ ,\ \boldsymbol{a}^{(l)} = \mathbf{W}^{(l)}\boldsymbol{h}^{(l-1)} + \boldsymbol{b}^{(l)}\ ,\ \boldsymbol{h}^{(l)} = \tau\left(\boldsymbol{a}^{(l)}\right)\ ,$$

for $l \in [L]$, where $\mathbf{W}^{(l)}$ and $\boldsymbol{b}^{(l)}$ are the weight matrix and bias vector of the l-th layer, respectively, τ is the nonlinear activation function, $[L]$ represents the set $\{1, 2, \cdots, L\}$, and $\mathbf{W}$ denotes the set of weight matrices, i.e., $\mathbf{W} = \{\mathbf{W}^{(l)}\}_{l=1}^{L}$. For notational simplicity, $\mathbf{W}$ is referred to as either a weight matrix or a set, depending on the context. We define the output of the DNN as $\hat{\boldsymbol{y}} = \boldsymbol{a}^{(L)}$. In this paper, we specify τ as the ReLU activation. For notational clarity, we will omit the explicit layer index l in the following discussion and use primed symbols to represent variables from the $(l-1)$-th layer, e.g., $\boldsymbol{h}' = \boldsymbol{h}^{(l-1)}$.

The objective of ternary quantization is to establish a ternary-weight DNN, provided a pretrained one. In the remainder of this paper, we denote $f(\boldsymbol{x};\mathbf{W})$ and $f(\boldsymbol{x};\mathbf{T})$ as Full-Precision DNNs (FP-DNNs) and Ternary DNNs (T-DNNs), respectively. Here, $\mathbf{T} = \{\mathbf{T}^{(l)}\}_{l=1}^{L}$, each element of which belongs to $\{-1, 0, 1\}$. Notice that $\mathbf{T}$ is referred to as either a weight matrix or a set, depending on the context.

Uncertainty Quantification. Similar to Bayesian-based uncertainty quantification methods, we also treat weights as random variables and make the following assumptions.

Assumption 1. *We assume that the weights within the same layer of FP-DNNs follow a Gaussian distribution, that is, $w^{(l)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu^{(l)}, (\sigma^{(l)})^2)$ for $l \in [L]$, where $w^{(l)}$ denotes an arbitrary element of matrix $\mathbf{W}^{(l)}$.*

Assumption 2. *We assume that the weights within the same layer of T-DNNs follow a ternary-valued distribution, that is, $t^{(l)} \overset{\text{i.i.d.}}{\sim} \mathcal{T}(p^{(l)}, 1 - p^{(l)} - q^{(l)}, q^{(l)})$ for $l \in [L]$, where $t^{(l)}$ denotes an arbitrary element of matrix $\mathbf{T}^{(l)}$ and $X \sim \mathcal{T}(p, 1-p-q, q)$ denotes that the random variable X follows a discrete distribution with $P(X = -1) = p$, $P(X = 0) = 1 - p - q$, and $P(X = 1) = q$ for $p, q \in [0, 1]$.*

Seminal studies (He and Fan 2019; Alawad and Ishak 2024; Zhang, Wang, and Fan 2024; Tian et al. 2025) made mild as-

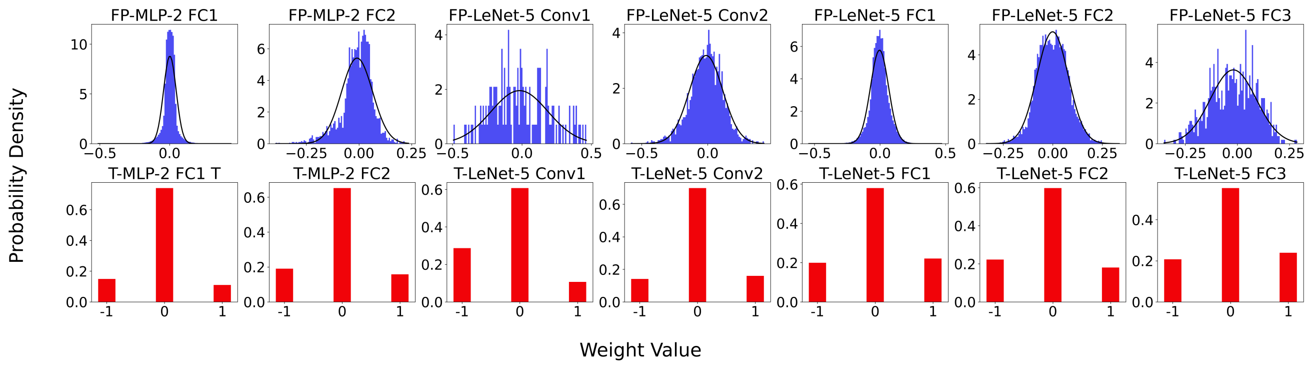


Figure 2: Empirical weight distributions of MLP-2 and LeNet-5 layer by layer.

sumptions similar to Assumption 1 for describing the weight distributions of FP-DNNs. We also conduct empirical investigations on the mildness of Assumptions 1 and 2. We first train a two-layer MLP (Rumelhart, Hinton, and Williams 1986) and a LeNet-5 (LeCun et al. 1998) with the SGD optimizer (Sutskever et al. 2013) on MNIST (LeCun et al. 1998), denoted as FP-MLP-2 and FP-LeNet-5, respectively. We then train their ternary-weight counterparts, i.e., T-MLP-2 and T-LeNet-5. Figure 2 shows the empirical weight distributions within the same layer for the concerned models, where the blue and red regions represent those of full-precision and ternary-value, respectively. It is observed that each blue region approximately follows a Gaussian distribution, hence supporting Assumption 1. In addition, it is obvious that each red region follows a ternary-valued distribution, hence supporting Assumption 2. Following this logic, we can estimate the parameters of weight distributions in Assumptions 1 and 2 from the empirical distributions.

3.2 Exact Moment Propagation

In this subsection, we introduce the EMP method. Figure 1(b) illustrates the EMP through a single layer. During feed-forward propagation, the concerned moments undergo two types of conversions. The linear transformation from $\boldsymbol{h}'$ to $\boldsymbol{a}$ induces the conversion from $(\boldsymbol{\mu}_{\boldsymbol{h}'}, \boldsymbol{\Sigma}_{\boldsymbol{h}'})$ to $(\boldsymbol{\mu}_{\boldsymbol{a}}, \boldsymbol{\Sigma}_{\boldsymbol{a}})$, which is formally expressed by the linear computation of EMP. Similarly, the nonlinear activation from $\boldsymbol{a}$ to $\boldsymbol{h}$ induces the conversion from $(\boldsymbol{\mu}_{\boldsymbol{a}}, \boldsymbol{\Sigma}_{\boldsymbol{a}})$ to $(\boldsymbol{\mu}_{\boldsymbol{h}}, \boldsymbol{\Sigma}_{\boldsymbol{h}})$, which is formally expressed by the nonlinear computation of EMP. The nonlinear computation of EMP calculates the covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{h}}$ element by element, where each element is obtained by solving an integral. The challenge lies in solving the integrals for the non-diagonal elements, as the function being integrated becomes more complex than that in the diagonal case (Frey and Hinton 1999). Previous studies have approximated these integrals by either constraining the integration intervals (Bibi, Alfadly, and Ghanem 2018) or using asymptotic methods (Wu et al. 2019). In contrast, we solve these integrals directly by applying a recursive approach based on integration by parts, leading to an exact solution for $\boldsymbol{\Sigma}_{\boldsymbol{h}}$. The following subsections elaborate on the completion of linear computation and nonlinear computation of the EMP.

Linear Computation of EMP. The goal of the linear computation of EMP is to provide exact expressions to calculate $(\boldsymbol{\mu}_{\boldsymbol{a}}, \boldsymbol{\Sigma}_{\boldsymbol{a}})$ using $(\boldsymbol{\mu}_{\boldsymbol{h}'}, \boldsymbol{\Sigma}_{\boldsymbol{h}'})$ and $(\boldsymbol{\mu}_{\mathbf{W}}, \boldsymbol{\Sigma}_{\mathbf{W}})$. We achieve this goal with the following theorem.

Theorem 3. *Let $\boldsymbol{h}'$ be a random vector characterized by moments $\boldsymbol{\mu}_{\boldsymbol{h}'}$ and $\boldsymbol{\Sigma}_{\boldsymbol{h}'}$, b_i denote a constant, and $\mathbf{W}_i$ be a random vector characterized by moments $\boldsymbol{\mu}_{\mathbf{W}_i}$ and $\boldsymbol{\Sigma}_{\mathbf{W}_i}$ for $i \in \{1,2\}$. Define the scalar random variable $a_i = \mathbf{W}_i^\top \boldsymbol{h}' + b_i$, where $i \in \{1,2\}$. Then, the following holds.*

$$
\begin{aligned}
&\mathbb{E}[a_i] = \boldsymbol{\mu}_{\mathbf{W}_i}^\top \boldsymbol{\mu}_{\boldsymbol{h}'} + b_i \,, \\
&\mathbb{V}(a_i) = \boldsymbol{\mu}_{\boldsymbol{h}'}^\top \boldsymbol{\Sigma}_{\mathbf{W}_i} \boldsymbol{\mu}_{\boldsymbol{h}'} + \operatorname{tr}\left(\boldsymbol{\Sigma}_{\mathbf{W}_i} \boldsymbol{\Sigma}_{\boldsymbol{h}'}\right) + \boldsymbol{\mu}_{\mathbf{W}_i}^\top \boldsymbol{\Sigma}_{\boldsymbol{h}'} \boldsymbol{\mu}_{\mathbf{W}_i} \,, \\
&\operatorname{Cov}(a_1, a_2) = \operatorname{tr}\left(\operatorname{Cov}(\mathbf{W}_1, \mathbf{W}_2) \boldsymbol{\Sigma}_{\boldsymbol{h}'}\right) + \boldsymbol{\mu}_{\mathbf{W}_1}^\top \boldsymbol{\Sigma}_{\boldsymbol{h}'} \boldsymbol{\mu}_{\mathbf{W}_2} \\
&\qquad\qquad + \boldsymbol{\mu}_{\boldsymbol{h}'}^\top \operatorname{Cov}(\mathbf{W}_1, \mathbf{W}_2) \boldsymbol{\mu}_{\boldsymbol{h}'} \,.
\end{aligned}
$$

Theorem 3 provides analytic expressions of the expectation and covariance of pre-activation variables. Notice that Theorem 3 is compatible with FP-DNNs and T-DNNs, leading to the following corollaries.

Corollary 4. *Under the conditions given in Theorem 3 and Assumption 1, the following holds.*

$$
\begin{aligned}
&\mathbb{E}[a_i] = \mu \mathbf{1}^\top \boldsymbol{\mu}_{\boldsymbol{h}'} + b_i \,, \\
&\mathbb{V}(a_i) = \sigma^2 \left(\|\boldsymbol{\mu}_{\boldsymbol{h}'}\|_2^2 + \operatorname{tr}\left(\boldsymbol{\Sigma}_{\boldsymbol{h}'}\right) \right) + \mu^2 \mathbf{1}^\top \boldsymbol{\Sigma}_{\boldsymbol{h}'} \mathbf{1} \,, \\
&\operatorname{Cov}(a_1, a_2) = \mu^2 \mathbf{1}^\top \boldsymbol{\Sigma}_{\boldsymbol{h}'} \mathbf{1} \,,
\end{aligned}
$$

where $\mathbf{1}$ denotes the vector with all components equal to 1 and $\|\cdot\|_2$ denotes the L_2 norm.

Corollary 5. *Under the conditions in Theorem 3 and Assumption 2, the following holds.*

$$
\begin{aligned}
&\mathbb{E}[a_i] = (q-p)\, \mathbf{1}^\top \boldsymbol{\mu}_{\boldsymbol{h}'} + b_i \,, \\
&\mathbb{V}(a_i) = \left[(q+p) - (q-p)^2 \right] \left(\|\boldsymbol{\mu}_{\boldsymbol{h}'}\|_2^2 + \operatorname{tr}\left(\boldsymbol{\Sigma}_{\boldsymbol{h}'}\right) \right) \\
&\qquad\qquad + (q-p)^2\, \mathbf{1}^\top \boldsymbol{\Sigma}_{\boldsymbol{h}'} \mathbf{1} \,, \\
&\operatorname{Cov}(a_1, a_2) = (q-p)^2\, \mathbf{1}^\top \boldsymbol{\Sigma}_{\boldsymbol{h}'} \mathbf{1} \,.
\end{aligned}
$$

Corollary 4 and Corollary 5 provide analytic expressions of the expectation and covariance of the pre-activation variable, avoiding the computation of the term $\operatorname{Cov}(\mathbf{W}_1, \mathbf{W}_2)$

in Theorem 3, thereby enabling a more efficient linear computation of EMP than that of Theorem 3. The full proof of Theorem 3 can be found in Appendix C.

Nonlinear Computation of EMP. The goal of the nonlinear computation of EMP is to provide exact expressions to calculate $(\boldsymbol{\mu_h}, \boldsymbol{\Sigma_h})$ using $(\boldsymbol{\mu_a}, \boldsymbol{\Sigma_a})$. The core is to characterize the pre-activation variable $\boldsymbol{a}$ according to the Law of the Unconscious Statistician (Feller 1991). In general, we make the following assumptions.

Assumption 6. *Under Assumption 1, we assume that the pre-activation variables of FP-DNNs follow a Gaussian distribution, that is,* $\boldsymbol{a}^{(l)} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{a}^{(l)}}, \boldsymbol{\Sigma}_{\boldsymbol{a}^{(l)}})$ *for* $l \in [L]$.

Assumption 7. *Under Assumption 2, we assume that the pre-activation variables of T-DNNs follow a Gaussian distribution, that is,* $\boldsymbol{a}^{(l)} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{a}^{(l)}}, \boldsymbol{\Sigma}_{\boldsymbol{a}^{(l)}})$ *for* $l \in [L]$.

We validate the mildness of Assumptions 6 and 7, and discuss the role and interdependencies of all assumptions made throughout this paper in Appendix A. Next, it suffices to analytically express $(\boldsymbol{\mu_h}, \boldsymbol{\Sigma_h})$ under the Gaussian assumption on $\boldsymbol{a}$. We separately denote the Probability Density Function (PDF) and the Cumulative Distribution Function (CDF) of the standard univariate Gaussian distribution as

$$\phi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \ , \ \Phi(x) = \int_{-\infty}^{x} \phi(u)\,\mathrm{d}u \ .$$

In addition, we separately denote the PDF and the CDF of the standard bivariate Gaussian distribution with correlation coefficient $\rho \in (-1, 1)$ as

$$\begin{cases} \phi_2(x_1, x_2; \rho) = \dfrac{1}{2\pi\sqrt{1-\rho^2}} \exp\left(\dfrac{x_1^2 - 2\rho x_1 x_2 + x_2^2}{-2(1-\rho^2)}\right) \ , \\ \Phi_2(x_1, x_2; \rho) = \displaystyle\int_{-\infty}^{x_1}\int_{-\infty}^{x_2} \phi_2(u_1, u_2; \rho)\,\mathrm{d}u_1\,\mathrm{d}u_2 \ . \end{cases}$$

With the notations above, we present the following theorem.

Theorem 8. *Given* $(a_1, a_2)^\top \sim \mathcal{N}_2(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ *with*

$$\boldsymbol{\mu} = (\mu_1, \mu_2)^\top \ , \ \boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix} \ ,$$

and $h_i = \tau(a_i)$ *for* $i \in \{1, 2\}$ *where* $\tau(\cdot)$ *denotes the ReLU activation, then the following holds.*

$$\begin{aligned} &\mathbb{E}[h_i] = \sigma_i\left(r_i\Phi(r_i) + \phi(r_i)\right) \ , \\ &\mathbb{V}(h_i) = \sigma_i^2\left[\left(1 + r_i^2\right)\Phi(r_i) + r_i\phi(r_i)\right] - \mathbb{E}[h_i]^2 \ , \\ &\mathrm{Cov}(h_1, h_2) = \sigma_1\sigma_2\big[r_2\phi(r_1)\Phi(c_2) + r_1\phi(r_2)\Phi(c_1) \\ &\quad + (r_1r_2 + \rho)\Phi_2(r_1, r_2; \rho) + \sqrt{1-\rho^2}\phi(r_2)\phi(c_1)\big] \\ &\quad - \mathbb{E}[Z_1]\mathbb{E}[Z_2] \ , \end{aligned}$$

in which $r_i = \mu_i/\sigma_i$, $c_i = (r_i - \rho r_{3-i})/\sqrt{1-\rho^2}$ *for* $i \in \{1, 2\}$, $\phi(\cdot)$ *and* $\Phi(\cdot)$ *denote the PDF and the CDF of the standard univariate normal distribution, respectively, and* $\Phi_2(\cdot, \cdot; \rho)$ *denotes the CDF of standard bivariate normal distribution with correlation coefficient* ρ.

Theorem 8 provides analytic expressions of expectation and covariance of $\tau(\boldsymbol{a})$, completing the nonlinear computation of EMP. Notice that Theorem 8 is straightforwardly compatible with both FP-DNNs and T-DNNs. To the best of our knowledge, Theorem 8 provides the first exact solution to the covariance $\boldsymbol{\Sigma_h}$ of a Gaussian vector passed through the ReLU activation under mild assumptions; in contrast, the previous methods provided either exact $\boldsymbol{\Sigma_h}$ under strong assumptions (Bibi, Alfadly, and Ghanem 2018) or approximated $\boldsymbol{\Sigma_h}$ with $\boldsymbol{h} = \tau(\boldsymbol{a})$ (Wu et al. 2019). The full proof of Theorem 8 can be found in Appendix D.

Algorithm 1: EMP for T-DNNs

Input: Sample distribution $\boldsymbol{\mu_x}, \boldsymbol{\Sigma_x}$, weight distribution parameters $\{p^{(l)},\ q^{(l)}\}_{l=1}^L$
Output: Output covariance $\boldsymbol{\Sigma_{\hat{y}}}$
Procedure:

```
1: Set $\boldsymbol{\mu}_{\boldsymbol{h}}^{(0)} = \boldsymbol{\mu_x}, \boldsymbol{\Sigma}_{\boldsymbol{h}}^{(0)} = \boldsymbol{\Sigma_x}$
2: for $1 \le l \le L$ do
3:     $(\boldsymbol{\mu}_{\boldsymbol{a}}^{(l)}, \boldsymbol{\Sigma}_{\boldsymbol{a}}^{(l)}) \Leftarrow (\boldsymbol{\mu}_{\boldsymbol{h}}^{(l-1)}, \boldsymbol{\Sigma}_{\boldsymbol{h}}^{(l-1)})$ and $(p^{(l)},\ q^{(l)})$ by Corollary 5
4:     $(\boldsymbol{\mu}_{\boldsymbol{h}}^{(l)}, \boldsymbol{\Sigma}_{\boldsymbol{h}}^{(l)}) \Leftarrow (\boldsymbol{\mu}_{\boldsymbol{a}}^{(l)}, \boldsymbol{\Sigma}_{\boldsymbol{a}}^{(l)})$ by Theorem 8
5: end for
6: $\boldsymbol{\Sigma_{\hat{y}}} = \boldsymbol{\Sigma}_{\boldsymbol{a}}^{(L)}$
```

To summarize the computations above, we can complete the EMP method for uncertainty quantification. Algorithm 1 lists the procedure of the EMP method for quantifying the uncertainty of T-DNNs, where Step 3 and Step 4 represent the linear computation and the nonlinear computation of the EMP, respectively. Algorithm 1 also adapts to FP-DNNs with two modifications: replacing the input $\{p^{(l)},\ q^{(l)}\}_{l=1}^L$ with $\{\mu^{(l)},\ \sigma^{(l)}\}_{l=1}^L$ and substituting Corollary 5 in Step 3 with Corollary 4.

4 Uncertainty Optimization

In this section, we propose an optimization method termed MOMent Alignment (MOMA) for improving the accuracy and reducing the uncertainty of T-DNNs. The MOMA is compatible with various quantization training algorithms for DNNs. Here, we also adopt ternary quantization as an example and draw the key ideas from Li et al. (2016) and Alemdar et al. (2017) to achieve ternary training. We detail the optimization procedure in Subsection 4.1, and then formally propose the corresponding MOMA in Subsection 4.2.

4.1 Ternary Training of T-DNNs

Following the supervised learning paradigm, the optimization problem of training a T-DNN f can be formulated as

$$\min_{\mathbf{T}} \mathcal{L}(\boldsymbol{y}, f(\boldsymbol{x}; \mathbf{T})) \ , \tag{1}$$

where $(\boldsymbol{x}, \boldsymbol{y})$ denotes the pair of input and target variables, and $\mathcal{L}(\cdot, \cdot)$ denotes the loss function. Let $s \in \mathbb{N}$ indicate the training iteration. The weight-updating procedure is listed as

$$\begin{cases} \mathbf{T}_{s+1} = \mathrm{sgn}_{\gamma_s}(\mathbf{W}_{s+1}) \ , \\ \mathbf{W}_{s+1} = \mathbf{W}_s + \eta\dfrac{\partial\mathcal{L}(\boldsymbol{y}, f(\boldsymbol{x}; \mathbf{T}_s))}{\partial\mathbf{W}_s} \ , \end{cases} \tag{2}$$

where η denotes the learning rate and $\text{sgn}_{\gamma_s}(\cdot)$ is the partition function. Here, we employ the partition function used in Li et al. (2016), that is,

$$t = \text{sgn}_{\gamma_s}(w) = \begin{cases} 1\,, & w > \gamma_s\,, \\ 0\,, & -\gamma_s \le w \le \gamma_s\,, \\ -1\,, & w < -\gamma_s\,, \end{cases} \tag{3}$$

where t and w denote an arbitrary element of the weight matrices $\mathbf{T}_s$ and $\mathbf{W}_s$, respectively, and $\gamma_s \in \mathbb{R}^+$ is a partition factor defined as

$$\gamma_s = \frac{1}{N}\left\|\mathbf{W}_s\right\|_1\ . \tag{4}$$

The gradient-related term $\partial\mathcal{L}(\boldsymbol{y}, f(\boldsymbol{x};\mathbf{T}_s))/\partial\mathbf{W}_s$ in Eq. (2) is computed via surrogate gradients (Bengio, Léonard, and Courville 2013) as

$$\begin{cases} \dfrac{\partial\mathcal{L}\left(\boldsymbol{y}, f\left(\boldsymbol{x};\mathbf{T}_s\right)\right)}{\partial\mathbf{W}_s} = \dfrac{\partial\mathcal{L}\left(\boldsymbol{y}, f\left(\boldsymbol{x};\mathbf{T}_s\right)\right)}{\partial\mathbf{T}_s}\dfrac{\partial\mathbf{T}_s}{\partial\mathbf{W}_s}\,, \\ \dfrac{\partial\mathbf{T}_s}{\partial\mathbf{W}_s} = \dfrac{\partial\text{sgn}_{\gamma}\left(\mathbf{W}_s\right)}{\partial\mathbf{W}_s} = 1\,. \end{cases} \tag{5}$$

In summary, one can obtain a T-DNN by initializing $\mathbf{W}_0$ with the pretrained full-precision weights, and following the optimization procedure listed in Eq. (2), Eq. (4), and Eq. (5).

4.2 Moment Alignment

This subsection proposes the MOMA optimization for improving the accuracy and reducing the uncertainty of T-DNNs, ternarized from a pretrained FP-DNN. The key idea of MOMA is to formulate an optimization problem by aligning the first and second central moments of T-DNN weights to zero. The following theorem supports our line of thought.

Theorem 9. *Under Assumption 2 and Assumption 7, if* $q - p = 0$ *and* $p + q - (q-p)^2 = 0$ *hold for the ternary-valued distribution parameters* $(p,\ q)$ *of each layer, then the output covariance satisfies* $\boldsymbol{\Sigma}_{\hat{\boldsymbol{y}}} = \mathbf{0}$.

Theorem 9 demonstrates that there exists an optimal solution, that is, the uncertainty of T-DNNs can be reduced to zero if one aligns the concerned moments to zero, which provides a theoretical guarantee for MOMA. The full proof of Theorem 9 can be found in Appendix E. Inspired by Theorem 9, we formulate the MOMA optimization as

$$\begin{cases} \tilde{\gamma}_s = \underset{\gamma}{\arg\min}\ \left\{\left|g_3\left(\gamma, \mathbf{W}_s\right)\right| + \lambda\left|g_4\left(\gamma, \mathbf{W}_s\right)\right|\right\}\,, \\ \text{s.t.}\quad \gamma \in \left(\left(1-c\right)\gamma_s, \left(1+c\right)\gamma_s\right)\ . \end{cases} \tag{6}$$

Algorithm 2: Ternary Training with MOMA for T-DNNs

Input: Pretrained full-precision weights $\mathbf{W}_0$, the maximum number of training epochs S_0
Output: Ternary weight $\mathbf{T}$
Procedure:

```
1: for $0 \le s \le S_0$ do
2:     $\mathbf{W}_{s+1} = \mathbf{W}_s + \eta\,\partial\mathcal{L}(\boldsymbol{y}, f(\boldsymbol{x};\mathbf{T}_s))/\partial\mathbf{T}_s$ in Eq. (2)
3:     $\gamma_s = \frac{1}{N}\|\mathbf{W}_s\|_1$ in Eq. (4)
4:     $\tilde{\gamma}_s \leftarrow$ solving Eq. (6)
5:     $\mathbf{T}_{s+1} = \text{sgn}_{\tilde{\gamma}_s}(\mathbf{W}_{s+1})$ in Eq. (3).
6: end for
```

where $c \in (0,1)$, $\lambda \in \mathbb{R}^+$, g_3 is an estimator for $q-p$, and g_4 is an estimator for $p+q-(q-p)^2$. The search interval is designed to be a neighborhood of the conventional partition factor of Eq. (4) to maintain accuracy, while the objective function is designed to align the concerned moments to zero for uncertainty reduction. The estimators in Eq. (6) are defined as follows.

$$\begin{cases} g_1\left(\gamma, \mathbf{W}_s\right) = \sum_{i,j}\chi_{(-\infty,-\gamma)}\left(\mathbf{W}_{ij}\right)/N\,, \\ g_2\left(\gamma, \mathbf{W}_s\right) = \sum_{i,j}\chi_{(\gamma,+\infty)}\left(\mathbf{W}_{ij}\right)/N\,, \\ g_3\left(\gamma, \mathbf{W}_s\right) = g_2\left(\gamma, \mathbf{W}_s\right) - g_1\left(\gamma, \mathbf{W}_s\right)\,, \\ g_4\left(r, \mathbf{W}_s\right) = g_1\left(\gamma, \mathbf{W}_s\right) + g_2\left(\gamma, \mathbf{W}_s\right) - \left(g_3\left(\gamma, \mathbf{W}_s\right)\right)^2\,, \end{cases}$$

where g_1 is an estimator for p, g_2 is an estimator for q, $\mathbf{W}_{ij}$ denotes the (i,j)-th element of $\mathbf{W}$, and $\chi_A(\cdot)$ denotes the indicator function of a concerned interval A that satisfies $\chi_A(x) = 1$ if $x \in A$ and $\chi_A(x) = 0$ if $x \notin A$.

Algorithm 2 details the calculation procedure of the MOMA optimization for T-DNN training. In contrast to conventional T-DNN training algorithms, MOMA involves an additional search for the partition factor, as detailed in Step 5. Hence, our proposed MOMA method can be incorporated into other quantization training algorithms that use a threshold-based partition function like Eq. (3).

5 Experiments

In this section, we conduct experiments to verify the effectiveness of the proposed EMP and MOMA methods. The experiments are performed to answer the following questions.

Q1. Is the proposed EMP method comparable to the classical uncertainty estimation methods?

Q2. To what extent, if at all, does weight ternarization increase the uncertainty of DNNs?

Q3. Whether and to what extent does the proposed MOMA method reduce the uncertainty of T-DNNs?

All experiments were conducted on MNIST (LeCun et al. 1998), CIFAR-10, and CIFAR-100 (Krizhevsky 2009) datasets. The concerned architectures include MLP 2[1] (Rumelhart, Hinton, and Williams 1986), LeNet-5 (LeCun et al. 1998), VGG-13 (Simonyan 2014), and ResNet-18 (He et al. 2016). We denote the full-precision DNNs as FP-DNNs, such as FP-MLP-2. The ternary-weight DNNs trained using the procedures outlined in Subsection 4.1 are referred to as T-DNNs, such as T-MLP-2. The ternary-weight DNNs trained with Algorithm 2 are denoted as T-DNNs-MOMA, such as T-MLP-2-MOMA. The proposed methods and contenders are implemented in `PyTorch` and `SciPy`. All experiments were conducted on a `13th gen Intel(R) Core(TM) i9-13900KF` CPU and an `NVIDIA RTX 4090` GPU. Experiments were repeated 5 times with different initialization seeds, reporting the mean α and standard deviation β as α_β.

[1]MLP-2 denotes a two-layer fully-connected architecture.

Datasets	o	FP-LeNet-5			T-LeNet-5		
		EMP	MP	PL-DNN	EMP	MP	PL-DNN
MNIST	0	1.0007 ± 0.0101	1.1809 ± 0.0217	1.2772 ± 0.0153	1.0023 ± 0.0109	–	–
	1	0.9999 ± 0.0103	1.1793 ± 0.0238	1.2759 ± 0.0139	1.0023 ± 0.0121	–	–
	2	0.9990 ± 0.0118	1.1794 ± 0.0239	1.2748 ± 0.0170	1.0021 ± 0.0117	–	–
	3	1.0008 ± 0.0104	1.1808 ± 0.0215	1.2770 ± 0.0153	1.0022 ± 0.0108	–	–
	4	1.0011 ± 0.0096	1.1807 ± 0.0239	1.2775 ± 0.0136	1.0023 ± 0.0107	–	–
	5	1.0010 ± 0.0106	1.1815 ± 0.0237	1.2773 ± 0.0149	1.0022 ± 0.0109	–	–
	6	1.0011 ± 0.0096	1.1818 ± 0.0246	1.2774 ± 0.0130	1.0023 ± 0.0105	–	–
	7	0.9996 ± 0.0099	1.1803 ± 0.0243	1.2755 ± 0.0143	1.0021 ± 0.0101	–	–
	8	1.0014 ± 0.0098	1.1814 ± 0.0218	1.2422 ± 0.0140	1.0021 ± 0.0107	–	–
	9	0.9999 ± 0.0100	1.1801 ± 0.0225	1.2759 ± 0.0142	1.0019 ± 0.0106	–	–
CIFAR-10	0	1.0098 ± 0.0099	1.5029 ± 0.0177	1.6167 ± 0.0334	1.0141 ± 0.0113	–	–
	1	1.0092 ± 0.0105	1.5020 ± 0.0207	1.6156 ± 0.0299	1.0136 ± 0.0115	–	–
	2	1.0090 ± 0.0127	1.5017 ± 0.0218	1.6154 ± 0.0358	1.0141 ± 0.0110	–	–
	3	1.0100 ± 0.0110	1.5031 ± 0.0189	1.6169 ± 0.0338	1.0141 ± 0.0115	–	–
	4	1.0106 ± 0.0098	1.5041 ± 0.0179	1.6179 ± 0.0307	1.0137 ± 0.0116	–	–
	5	1.0099 ± 0.0109	1.5030 ± 0.0200	1.6167 ± 0.0311	1.0139 ± 0.0118	–	–
	6	1.0103 ± 0.0094	1.5036 ± 0.0185	1.6174 ± 0.0282	1.0139 ± 0.0109	–	–
	7	1.0090 ± 0.0099	1.5017 ± 0.0184	1.6153 ± 0.0307	1.0139 ± 0.0120	–	–
	8	1.0103 ± 0.0093	1.5036 ± 0.0179	1.6173 ± 0.0312	1.0140 ± 0.0118	–	–
	9	1.0095 ± 0.0100	1.5025 ± 0.0179	1.6162 ± 0.0314	1.0141 ± 0.0109	–	–

Table 1: Performance of the EMP and contenders for LeNet-5 on MNIST and CIFAR-10. Results for other architectures and datasets are detailed in Appendix F.

5.1 Verifications on EMP

This subsection demonstrates the effectiveness of the proposed EMP method. Let $\boldsymbol{x} \in D$ denote a test sample from the test set D. In Algorithm 1, we set $\boldsymbol{\mu}_{\boldsymbol{x}} = \boldsymbol{x}$ and $\boldsymbol{\Sigma}_{\boldsymbol{x}} = \mathbf{0}$.

To compare the performance of various uncertainty estimators, we follow the evaluation metric in Bibi, Alfadly, and Ghanem (2018), where they treat the classical Monte Carlo (MC) estimator (Jospin et al. 2022) as the ground truth, with other estimators being evaluated based on their closeness to it. Formally, we define the variance ratio as $r_o(\boldsymbol{x}) = \mathbb{V}_o^{\mathrm{MC}}(\boldsymbol{x}) \ / \ \mathbb{V}_o^{\mathrm{E}}(\boldsymbol{x})$, where $\mathbb{V}_o^{\mathrm{MC}}(\boldsymbol{x})$ and $\mathbb{V}_o^{\mathrm{E}}(\boldsymbol{x})$ denote the o-th element of the output variance vector obtained by the MC method and by an estimator, respectively, for $o \in \{0, 1, \cdots, 9\}$. A ratio $r_o(\boldsymbol{x})$ closer to 1 indicates higher precision of an estimator on $\boldsymbol{x}$. We then compute the mean and standard deviation of $r_o(\boldsymbol{x})$ over D as

$$\mathbb{E}\left[r_o\left(\boldsymbol{x}\right)\right] = \sum\nolimits_{\boldsymbol{x}\in D} r_o\left(\boldsymbol{x}\right) / \left|D\right| \ ,$$
$$\sqrt{\mathbb{V}\left[r_o\left(\boldsymbol{x}\right)\right]} = \sqrt{\sum\nolimits_{\boldsymbol{x}\in D} \left(r_o\left(\boldsymbol{x}\right) - \mathbb{E}\left[r_o\left(\boldsymbol{x}\right)\right]\right)^2 / \left|D\right|} \ .$$

Then we use $\mathbb{E}[r_o(\boldsymbol{x})] \pm \sqrt{\mathbb{V}[r_o(\boldsymbol{x})]}$ as a metric, which is the same as that in Bibi, Alfadly, and Ghanem (2018), to measure the uncertainty estimation performance of an estimator. A mean $\mathbb{E}[r_o(\boldsymbol{x})]$ closer to 1 indicates higher precision on average, while a standard deviation $\sqrt{\mathbb{V}[r_o(\boldsymbol{x})]}$ closer to 0 indicates greater robustness on average.

Table 1 shows the uncertainty estimation performance of EMP and its contenders, including PL-DNN (Bibi, Alfadly, and Ghanem 2018) and MP (Wu et al. 2019) for LeNet-5 on MNIST and CIFAR-10. It is observed that the EMP method achieves near-ideal performance and significantly outperforms its contenders. Experimental results for other architectures and datasets, detailed in Appendix F, further support this observation. These findings demonstrate that the proposed EMP estimator is comparable to the MC estimator and more precise than its contenders, thus answering Q1.

5.2 Verifications on MOMA

This experiment demonstrates the effectiveness of the MOMA method. We first analyze the uncertainty difference between DNNs with and without weight ternarization to take a closer look at Q2. We begin this analysis by defining two variance ratios as

$$\tilde{r}_o\left(\boldsymbol{x}\right) = \mathbb{V}_o^{\mathrm{E}}(\boldsymbol{x}; \text{T-DNN}) \ / \ \mathbb{V}_o^{\mathrm{E}}(\boldsymbol{x}; \text{FP-DNN}) \ ,$$
$$\tilde{r}'_o\left(\boldsymbol{x}\right) = \mathbb{V}_o^{\mathrm{E}}(\boldsymbol{x}; \text{T-DNN-MOMA}) \ / \ \mathbb{V}_o^{\mathrm{E}}(\boldsymbol{x}; \text{FP-DNN}) \ ,$$

where $\mathbb{V}_o^{\mathrm{E}}(\boldsymbol{x}; \text{Model})$ represents the variance, i.e., $\mathbb{V}_o^{\mathrm{E}}(\boldsymbol{x})$, computed using the corresponding model. A ratio greater than 1 indicates that weight ternarization increases DNN uncertainty by the corresponding ratio.

Figure 3 shows the concerned uncertainty difference on MLP-2 by providing the histogram of the two ratios. Due to space limitations, we only plot the results for $o \in \{0, 1, \cdots, 5\}$ and the ratio range of $[0, 3000]$, containing 99% of the empirical points for the variance ratios. Full results are given in Appendix G. The blue and red regions represent the histograms of $\tilde{r}_o(\boldsymbol{x})$ and $\tilde{r}'_o(\boldsymbol{x})$, respectively. It is observed that $\tilde{r}_o(\boldsymbol{x}) > 1.4$ and $\tilde{r}'_o(\boldsymbol{x}) > 1.4$ for all

Models	Accuracy	Uncertainty	Reduction
FP-MLP-2	$0.9761_{0.0002}$	$1.4697_{0.0082} \times 10^3$	
T-MLP-2	$0.9752_{0.0005}$	$1.2572_{0.0236} \times 10^5$	$68.08_{0.1389}\%$
T-MLP-2-MOMA	$0.9751_{0.0006}$	$4.0129_{0.0039} \times 10^4$	
FP-LeNet-5	$0.9911_{0.0001}$	$1.7728_{0.0232} \times 10^3$	
T-LeNet-5	$0.9903_{0.0003}$	$2.8113_{0.0490} \times 10^5$	$98.55_{0.0025}\%$
T-LeNet-5-MOMA	$0.9885_{0.0006}$	$4.0906_{0.1433} \times 10^3$	
FP-VGG-13	$0.9985_{0.0001}$	$4.6029_{0.2302} \times 10^{-1}$	
T-VGG-13	$0.9954_{0.0005}$	$1.4500_{0.0210} \times 10^1$	$20.86_{2.5366}\%$
T-VGG-13-MOMA	$0.9957_{0.0005}$	$1.1463_{0.0198} \times 10^1$	
FP-ResNet-18	$0.9963_{0.0001}$	$6.9757_{0.3743} \times 10^{-3}$	
T-ResNet-18	$0.9938_{0.0002}$	$4.2808_{0.1772} \times 10^{-1}$	$44.27_{4.4212}\%$
T-ResNet-18-MOMA	$0.9951_{0.0002}$	$2.4067_{0.0501} \times 10^{-1}$	

Table 2: Accuracy and uncertainty of concerned models on MNIST. Results for other datasets are detailed in Appendix G.

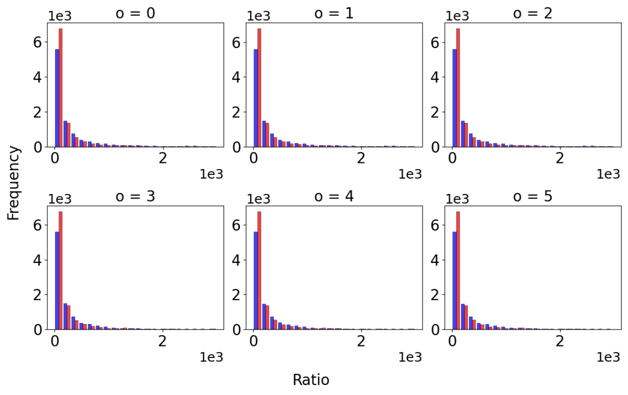


Figure 3: Histograms of the concerned variance ratios.

$x \in D$, indicating that DNN uncertainty increases after weight ternarization, regardless of whether MOMA is used. This observation answers Q2. Furthermore, the red bars are taller at lower x-values and shorter at higher x-values, indicating that the proposed MOMA method effectively reduces the concerned variance ratio.

To further measure the uncertainty reduction performance of the proposed MOMA method, we define the uncertainty of a model on a dataset D as

$$\text{Uncertainty} = \frac{1}{|D|} \sum_{x \in D} \operatorname{tr}\left(\Sigma_{\hat{y}(x)}\right) .$$

Table 2 shows the uncertainty reduction performance of MOMA across various DNN architectures on MNIST by comparing the uncertainty of the T-DNNs and T-DNNs-MOMA. It is observed that the MOMA greatly reduces the uncertainty of T-DNNs on MNIST. Experimental results for CIFAR-10 and CIFAR-100, provided in Appendix G, further support this observation. This finding demonstrates that the proposed MOMA can effectively reduce the uncertainty of T-DNNs, thus answering Q3.

6 Conclusions

In this paper, we proposed the EMP, a theoretical uncertainty quantification method that works independently of weight precision. Experimental results showed that T-DNNs exhibit significantly greater uncertainty than FP-DNNs across varying ratios. Inspired by the EMP estimator, we proposed MOMA, an optimization method for reducing the uncertainty of T-DNNs. Empirical results validated the effectiveness of both EMP and MOMA.

Acknowledgements

This research was supported by the Jiangsu Provincial Natural Science Foundation Youth Project (BK20230782) and Nanjing University-China Mobile Communications Group Co., Ltd. Joint Institute.

References

Alawad, M.; and Ishak, M. 2024. Probabilistic Bayesian Neural Networks for Efficient Inference. In *Proceedings of the 34th Great Lakes Symposium on VLSI*, 724–729.

Alemdar, H.; Leroy, V.; Prost-Boucle, A.; and Pétrot, F. 2017. Ternary neural networks for resource-efficient AI applications. In *Proceedings of the 30th International Joint Conference on Neural Networks*, 2547–2554.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Bibi, A.; Alfadly, M.; and Ghanem, B. 2018. Analytic expressions for probabilistic moments of PL-DNN with Gaussian input. In *Proceedings of the 31st IEEE Conference on Computer Vision and Pattern Recognition*, 9099–9107.

Courbariaux, M.; Bengio, Y.; and David, J.-P. 2015. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in Neural Information Processing Systems 28*, 3123–3131.

Einbinder, B.-S.; Romano, Y.; Sesia, M.; and Zhou, Y. 2022. Training uncertainty-aware classifiers with conformalized deep learning. In *Advances in Neural Information Processing Systems 35*, 22380–22395.

Feller, W. 1991. *An introduction to probability theory and its applications*. John Wiley & Sons.

Frey, B.; and Hinton, G. 1999. Variational learning in nonlinear Gaussian belief networks. *Neural Computation*, 11(1): 193–213.

Gope, D.; Dasika, G.; and Mattina, M. 2019. Ternary hybrid neural-tree networks for highly constrained iot applications. *Proceedings of Machine Learning and Systems*, 1: 190–200.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

He, Z.; and Fan, D. 2019. Simultaneously optimizing weight and quantizer of ternary neural network using truncated gaussian approximation. In *Proceedings of the 32nd IEEE Conference on Computer Vision and Pattern Recognition*, 11438–11446.

Hubara, I.; Courbariaux, M.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Binarized neural networks. *Advances in Neural Information Processing Systems 29*, 4107–4115.

Jospin, L.-V.; Laga, H.; Boussaid, F.; Buntine, W.; and Bennamoun, M. 2022. Hands-on Bayesian neural networks—A tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2): 29–48.

Krizhevsky, A. 2009. Learning multiple layers of features from tiny images. *Master's thesis*.

Lakshminarayanan, B.; Pritzel, A.; and Blundell, C. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st Advances in Neural Information Processing Systems 30*, 6405–6416.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Li, F.; Liu, B.; Wang, X.; Zhang, B.; and Yan, J. 2016. Ternary weight networks. *arXiv preprint arXiv:1605.04711*.

Li, Y.; Xu, S.; Zhang, B.; Cao, X.; Gao, P.; and Guo, G. 2022. Q-vit: Accurate and fully quantized low-bit vision transformer. In *Advances in neural information processing systems 35*, 34451–34463.

Lin, Z.; Courbariaux, M.; Memisevic, R.; and Bengio, Y. 2016. Neural networks with few multiplications. In *Proceedings of the 4th International Conference on Learning Representations*.

Meng, X.; Bachmann, R.; and Khan, M.-E. 2020. Training binary neural networks using the Bayesian learning rule. In *Proceedings of the 37th International Conference on Machine Learning*, 6852–6861.

Peters, J.; and Welling, M. 2018. Probabilistic binary neural networks. *arXiv preprint arXiv:1809.03368*.

Rastegari, M.; Ordonez, V.; Redmon, J.; and Farhadi, A. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *Proceedings of the 14th European Conference on Computer Vision*, 525–542.

Rumelhart, D.; Hinton, G.; and Williams, R. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088): 533–536.

Shen, X.; Dong, P.; Lu, L.; Kong, Z.; Li, Z.; Lin, M.; Wu, C.; and Wang, Y. 2024. Agile-Quant: Activation-Guided Quantization for Faster Inference of LLMs on the Edge. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, 18944–18951.

Simonyan, K. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sutskever, I.; Martens, J.; Dahl, G.; and Hinton, G. 2013. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning*, 1139–1147.

Tabarisaadi, P.; Khosravi, A.; Nahavandi, S.; Shafie-Khah, M.; and Catalão, J. 2022. An optimized uncertainty-aware training framework for neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 35(5): 6928–6935.

Tian, Y.-M.; Liang, S.; Zhang, S.-Q.; and Fan, F.-L. 2025. Depth-induced NTK: Bridging Over-parameterized Neural Networks and Deep Neural Kernels. *arXiv preprint arXiv:2511.05585*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems 30*, 5998–6008.

Wu, A.; Nowozin, S.; Meeds, E.; Turner, R.; Hernandez-Lobato, J.; and Gaunt, A. 2019. Deterministic variational inference for robust Bayesian neural networks. In *Proceedings of the 7th International Conference on Learning Representations*.

Zhang, S.-H.; Tang, W.-C.; Wu, C.; Hu, P.; Li, N.; Zhang, L.-J.; Zhang, Q.; and Zhang, S.-Q. 2025. TernaryCLIP: Efficiently Compressing Vision-Language Models with Ternary Weights and Distilled Knowledge. *arXiv:2510.21879*.

Zhang, S.-Q.; Wang, F.; and Fan, F.-L. 2024. Neural Network Gaussian Processes by Increasing Depth. *IEEE Transactions on Neural Networks and Learning Systems*, 35(2): 2881–2886.

Zhang, W.; Hou, L.; Yin, Y.; Shang, L.; Chen, X.; Jiang, X.; and Liu, Q. 2020. Ternarybert: Distillation-aware ultra-low bit bert. In *Proceedings of the 25th Empirical Methods in Natural Language Processing*, 509–521.

Zhu, C.; Han, S.; Mao, H.; and Dally, W. 2017. Trained ternary quantization. In *Proceedings of the 5th International Conference on Learning Representations*, 1–10.