For question (1), the correlation among weights in kernels was indeed considered in the early design of MetaQuant: the meta gradient of weights is determined by its surrounding weights and the gradient of its quantized weight. However, such an implementation showed roughly 10-20% drop in performance compared to the current design. Besides, the kernel sizes differ across layers in a deep neural network: a meta quantizer that receives $3 \times 3$ kernels as inputs cannot be applied to the layers with $1 \times 1$ kernels. The independent process for each weight in the current design endows MetaQuant with generalization to arbitrary architectures of neural networks.

For question (2), we further tested training time per iteration as suggested for MetaQuant and DoReFa with STE using ResNet20 in CIFAR10 (Intel Xeon CPU E5-1650 with GeForce GTX 750 Ti). MetaQuant costs 51.15 seconds to finish one iteration of training while baseline method uses 38.17s. We will add this training time analysis in final version.

For question (3), we further conducted experiments and added some state-of-the-art results of binary / ternary network on ImageNet in Table 1:

<table>
<thead>
<tr>
<th>Network</th>
<th>Method</th>
<th>bits</th>
<th>Top 1/5 drop (%)</th>
<th>Network</th>
<th>Method</th>
<th>bits</th>
<th>Top 1/5 drop (%)</th>
<th>Network</th>
<th>Method</th>
<th>bits</th>
<th>Top 1/5 drop (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet18</td>
<td>MetaQuant</td>
<td>1</td>
<td>6.32/4.13</td>
<td>ResNet18</td>
<td>MetaQuant</td>
<td>1</td>
<td>5.17/3.59</td>
<td>STE</td>
<td>1</td>
<td>3.55/2.65</td>
<td></td>
</tr>
<tr>
<td>MobileNetV2</td>
<td>MetaQuant</td>
<td>4</td>
<td>7.70/5.43</td>
<td>MobileNetV2</td>
<td>ELQ</td>
<td>1</td>
<td>6.29/4.58</td>
<td>MobileNetV2</td>
<td>STE</td>
<td>4</td>
<td>3.71/1.89</td>
</tr>
</tbody>
</table>

Table 1: Comparison experiments in ImageNet. ‘elongated’ methods in paper that use dorefa as forward and STE as backward. ‘∗’: ELQ is a combination of a series of previous quantization methods and tracks on incremental quantization. MetaQuant focuses more on how to improve STE-based training quantization, without any extra loss and training tricks. ‘∗∗’: TTQ is a non-symmetric ternarization with $\{0, \alpha, -\beta\}$ as ternary points. MetaQuant follows dorefa using a symmetric quantization which leads to efficient inference.

To Reviewer 2: Regarding “in eq. (8) the term $W_\ast$ ...”, we would like to clarify the computation order in a forward pass.

In fact, for an iteration $t$, $\hat{W}_t$ is computed after $\phi$ as shown in the computation graph in Fig. 2 of the paper. A more detailed illustration is shown in Fig. 1. The meta quantizer $M_\phi$ takes $\partial L / \partial W_{t-1}$ and $\hat{W}_{t-1}$ in the previous iteration as inputs to compute its output, parameterized by $\phi$. The output of meta quantizer is then added to $\hat{W}_{t-1}$ to generate $\hat{W}_t$ in the current iteration $t$. Therefore, $\partial \hat{W} / \partial \phi$ can be computed if we track the path from $\phi$ to $\hat{W}_t$.

Regarding “... there seems to be a chicken-egg problem”, the meta quantizer is actually linked to the final loss $L$ of the base network with the following computation path: $\phi \rightarrow \Delta W \rightarrow W \rightarrow \hat{W} \rightarrow L$, according to Fig. 1. In detail: Step 1: In iteration $t$, $\partial L / \partial W_{t-1}$ and $\hat{W}_{t-1}$ (noted they are from the previous iteration) are fed into meta quantizer as data to generate meta gradient $\Delta W$. Step 2: $\Delta W$ contributes to $W_{t}$, which is quantized to $\hat{W}_{t}$. Step 3: $\hat{W}_{t}$ is involved into convolution or fully connected operation with input features from the base network, finally leads to the loss.

Intuitively, we can regard $\partial L / \partial \hat{W}_{t-1}$ and $\hat{W}_{t-1}$ from previous iteration as data to meta quantizer (\phi) for generating a component of $\hat{W}_t$, and this process is differentiable. We can get $\partial \hat{W} / \partial \phi$ using backpropagation, which can be passed to $\phi$ by chain rules.

For the writing issues, thanks for pointing then out. We will correct them in the final version.

To Reviewer 3: For your comments. In fact, we conducted experiments in ImageNet in Table 3 on page 7 in the original submission. Here, we have further conducted experiments, and added more state-of-the-art results of binary / ternary network on ImageNet in Table 1 for comparison.

References
