Learning Channel-wise Interactions for Binary Convolutional Neural Networks: Supplementary Material

Ziwei Wang, *Student Member, IEEE,* Jiwen Lu, *Senior Member, IEEE,* and Jie Zhou, *Senior Member, IEEE*

Appendix A
The REINFORCE algorithm to Optimize the Policy Network

The objective to learn the optimal policy network is maximizing the expected return over the entire CI-BCNN learning process:

$$\max_{\theta} Z(\theta) = \mathbb{E}_{\pi}[\sum_{\tau=1}^{T} \gamma r(s_\tau, a_\tau, s_{\tau+1})] \tag{1}$$

where θ means parameters in the policy network and π represents the selected policy. T stands for the time of sampling for each training batch and γ is the discount factor. According to the policy gradient method, we compute the expected gradient of the objective as follows:

$$\nabla_\theta Z = -\mathbb{E}_{\pi}[r(s_\tau, a_\tau, s_{\tau+1}) \nabla_\theta \log p(a_\tau|s_\tau)] \tag{2}$$

where the discount factor is combined into $r(s_\tau, a_\tau, s_{\tau+1})$ for simplicity. We apply Monte-Carlo Sampling to obtain the approximated gradients due to the intractability of exhaustion for all possible states. Sequence of states and actions $(s_1, a_1; ...; s_{T_k}, a_{T_k})$ are sampled M times in each training epoch:

$$\nabla_\theta Z = -\frac{1}{M}\sum_{k=1}^{M} R_k \sum_{\tau=0}^{T_k} \nabla_\theta \log p(a_\tau|s_\tau) \tag{3}$$

where T_k and R_k represent the number of steps and gained reward in the k_{th} sequence respectively. However, $p(a_\tau|s_\tau)$ in (3) is entangled by actions for exploring edge existence and influence, and the probability to choose influence is deterministic and non-differentiable. In order to back-propagate gradients, we approximate the optimization problem in a disentangle and differentiable manner. It is apparent the existence and influence are independent, so we rewrite the gradient:

$$\nabla_\theta Z = -\mathbb{E}_{\pi}[r_\tau(s_\tau, a_\tau)(\nabla_\theta \log p(a_\tau^e|s_\tau) + \nabla_\theta \log p(a_\tau^f|s_\tau))]$$

where a_τ^e means the action to change existence of the graph i.e. creating edges, deleting edges and keeping edges unchanged and a_τ^f represents the choice of parameters for influence i.e. the value of K_{ts}^l.

As for the former action probability of existence, we denote $-\mathbb{E}_{\pi}[r_\tau(s_\tau, a_\tau)\nabla_\theta \log p(a_\tau^e|s_\tau)]$ as L^e and rewrite it as follows:

$$\begin{aligned}
L^e = & -\mathbb{E}_{\pi}[r_\tau(s_\tau, a_\tau)((\nabla_\theta \log p(a_\tau^{e,c}|s_\tau) + \\
& \nabla_\theta \log p(a_\tau^{e,d}|s_\tau, a_\tau^{e,c}) + \nabla_\theta \log p(a_\tau^{e,u}|s_\tau, a_\tau^{e,c}, a_\tau^{e,d}))] \\
& = -\mathbb{E}_{\pi}[r_\tau(s_\tau, a_\tau)((\frac{\partial logp(a_\tau^{e,c}|s_\tau)}{\partial W_{ea}^l}\frac{\partial W_{ea}^l}{\partial \theta} + \\
& \frac{\partial logp(a_\tau^{e,d}|s_\tau, a_\tau^{e,c})}{\partial W_{ea}^{\prime l}}\frac{\partial W_{ea}^{\prime l}}{\partial W_{ea}^l}\frac{\partial W_{ea}^l}{\partial \theta} + 0))] \\
& = -\mathbb{E}_{\pi}[r_\tau(s_\tau, a_\tau)((\frac{\partial logp(a_\tau^{e,c}|s_\tau)}{\partial W_{ea}^l}\frac{\partial W_{ea}^l}{\partial \theta} + \\
& \alpha\frac{\partial logp(a_\tau^{e,d}|s_\tau, a_\tau^{e,c})}{\partial W_{ea}^{\prime l}} inv(W_{ea}^l)\frac{\partial W_{ea}^l}{\partial \theta}))]
\end{aligned} \tag{4}$$

where $inv(W)$ means a matrix whose elements are reciprocal with those in W and α is a small hyperparameter to approximate the derivative of normalizing $W_{ea}^{'l}$ because it is computationally expensive.

Since the sampling strategy for influence is deterministic, we apply a symmetric clip function to approximate the integral of delta distribution. At first, we explicitly write the probability of actions for influence selection:

$$p(a_\tau^f|s_\tau) = \begin{cases} \int_{-\frac{|a_\tau^f|-1}{2K_0}}^{-\frac{|a_\tau^f|-3}{2K_0}} \delta(t - W_{fa}^l)dt & a_\tau^f < 0 \\ \int_{\frac{|a_\tau^f|-3}{2K_0}}^{\frac{|a_\tau^f|-1}{2K_0}} \delta(t - W_{fa}^l)dt & a_\tau^f > 0 \end{cases} \tag{5}$$

- *Z. Wang and J. Lu are with the State Key Lab of Intelligent Technologies and Systems, Beijing National Research Center for Information Science and Technology (BNRist) and the Department of Automation, Tsinghua University, Beijing 100084, China.*
  E-mail: wang-zw18@mails.tsinghua.edu.cn, lujiwen@tsinghua.edu.cn.
- *J. Zhou is with the State Key Lab of Intelligent Technologies and Systems, Beijing National Research Center for Information Science and Technology (BNRist), the Department of Automation, Tsinghua University, Beijing 100084, China, and also with the Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China.*
  E-mail: jzhou@tsinghua.edu.cn

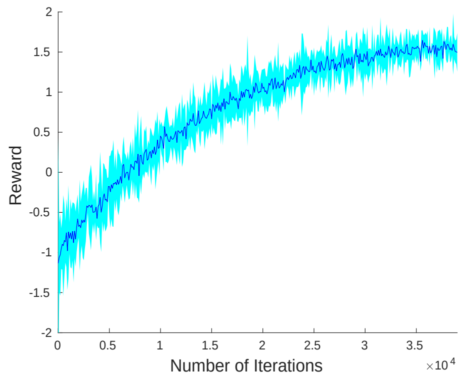

Figure 1. Average reward over five random seeds on CIFAR10 with the VGG16 architecture. The solid line demonstrates the average reward and the shaded area depicts the mean $\pm$ the standard deviation.

It is apparent that $\log p(a_\tau^f|s_\tau)$ is non-differentiable of W_{fa}^l. For any element $w_{fa}^l \in W_{fa}^l$, we use the following function to substitute original objective:

$$p(a_\tau^f|s_\tau) \approx \begin{cases} 2K_0 w_{fa}^l - a_\tau^f + 3 & w_{fa}^l \in (\frac{a_\tau^f - 3}{2K_0}, \frac{a_\tau^f - 2}{2K_0}] \cup w_{fa}^l \geqslant 0 \\ -2K_0 w_{fa}^l + a_\tau^f - 1 & w_{fa}^l \in (\frac{a_\tau^f - 2}{2K_0}, \frac{a_\tau^f - 1}{2K_0}] \cup w_{fa}^l \geqslant 0 \\ 2K_0 w_{fa}^l - a_\tau^f - 1 & w_{fa}^l \in (\frac{a_\tau^f + 1}{2K_0}, \frac{a_\tau^f + 2}{2K_0}] \cup w_{fa}^l < 0 \\ -2K_0 w_{fa}^l + a_\tau^f + 3 & w_{fa}^l \in (\frac{a_\tau^f + 2}{2K_0}, \frac{a_\tau^f + 3}{2K_0}] \cup w_{fa}^l < 0 \end{cases} \tag{6}$$

We can rewrite the gradient for updating the parameters in the policy network to learn the influence in REINFORCE algorithm. We use L_i to substitute $-\mathbb{E}_\pi[r_\tau(s_\tau, a_\tau)\nabla_\theta \log p(a_\tau^f|s_\tau)]$ as follows:

$$L_i = -\mathbb{E}_\pi[r_\tau(s_\tau, a_\tau) \cdot 2K_0 \cdot Sn(w_{fa}^l - \frac{|a_\tau^f| - 2}{2K_0} Sn(a_\tau^f)) \tag{7}$$

where the logarithm in the original gradient is removed and Sn represents the sign function for simplicity.

In summary, the approximation of the expected gradient can be written as follows:

$$\nabla_\theta Z_{approx} = -\frac{1}{M}\sum_{k=1}^{M} R_k \sum_{\tau=0}^{T_k}[\frac{\partial logp(a_\tau^{e,c}|s_\tau)}{\partial W_{ea}^l}\frac{\partial W_{ea}^l}{\partial \theta} + 2K_0 \cdot$$
$$Sn(w_{fa}^l - \frac{|a_\tau^f| - 2}{2K_0} Sn(a_\tau^f)) + \alpha \frac{\partial logp(a_\tau^{e,d}|s_\tau, a_\tau^{e,c})}{\partial W_{ea}'^l} inv(W_{ea}^l)\frac{\partial W_{ea}^l}{\partial \theta}] \tag{8}$$

Appendix B
Reward Curves on CIFAR-10

In order to prove the policy network in CI-BCNN learns to optimize the right objectives, we also plot the reward averaged over five random seeds. We conducted the experiments on CIFAR-10 with the VGG16 architecture. Figure 1 illustrates the average reward in the solid line, and the shaded area depicts the mean $\pm$ the standard deviation. Our method effectively learns the channel-wise interactions via the reward as the reward curves increase steadily and converge at the end of the training stage.