SUPPLEMENTARY DOCUMENT

Mirror Descent View for Neural Network Quantization

Here, we first provide the proofs and the technical derivations. Later we give additional experiments and the details of our experimental setting.

A MD Proofs and Derivations

A.1 More details on Projections

Here, we provide more details on projections used in the paper for Neural Network (NN) quantization. Even though we consider differentiable projections, Theorem 1 in the main paper does not require the projection to be differentiable. For the rest of the section, we assume $m = 1$, *i.e.*, consider projections that are independent for each $j \in \{1, \dots, m\}$.

Example 1 (**w**-space, binary, tanh)**.** Consider the tanh function, which projects a real value to the interval $[-1, 1]$:

$$w = P_{\beta_k}(\tilde{w}) := \tanh(\beta_k \tilde{w}) = \frac{\exp(2\beta_k \tilde{w}) - 1}{\exp(2\beta_k \tilde{w}) + 1} \ , \tag{1}$$

where $\beta_k \geq 1$ is the annealing hyperparameter and when $\beta_k \to \infty$, tanh approaches the step function. The inverse of the tanh is:

$$P^{-1}_{\beta_k}(w) = \frac{1}{\beta_k} \tanh^{-1}(w) = \frac{1}{2\beta_k} \log \frac{1+w}{1-w} \ . \tag{2}$$

Note that, $P^{-1}_{\beta_k}$ is monotonically increasing for a fixed β_k. Correspondingly, the mirror map from Theorem 1 in the main paper can be written as:

$$\begin{aligned} \Phi_{\beta_k}(w) &= \int P^{-1}_{\beta_k}(w) dw \\ &= \frac{1}{2\beta_k} \big[(1+w)\log(1+w) + (1-w)\log(1-w) \big] \ . \end{aligned} \tag{3}$$

Here, the constant from the integration is ignored. It can be easily verified that $\Phi_{\beta_k}(w)$ is in fact a valid mirror map. Correspondingly, the Bregman divergence can be written as:

$$\begin{aligned} D_{\Phi_{\beta_k}}(w, v) &= \Phi_{\beta_k}(w) - \Phi_{\beta_k}(v) - \Phi'_{\beta_k}(v)(w - v) \ , \quad \text{where } \Phi'_{\beta_k}(v) = \tfrac{1}{2\beta_k} \log \tfrac{1+v}{1-v} \ , \\ &= \frac{1}{2\beta_k} \left[w \log \frac{(1+w)(1-v)}{(1-w)(1+v)} + \log(1-w)(1+w) - \log(1-v)(1+v) \right] \ . \end{aligned} \tag{4}$$

Now, consider the proximal form of Mirror Descent (MD) update

$$w^{k+1} = \operatorname*{argmin}_{\mathbf{x} \in (-1,1)} \langle \eta \, g^k, w \rangle + D_{\Phi_{\beta_k}}(w, w^k) \ . \tag{5}$$

The idea is to find w such that the KKT conditions are satisfied. To this end, let us first write the Lagrangian of Eq. (5) by introducing dual variables y and z corresponding to the constraints $w > -1$ and $w < 1$, respectively:

$$\begin{aligned} F(w, x, y) &= \eta g^k w + y(-w - 1) + z(w - 1) \\ &\quad + \frac{1}{2\beta_k} \left[w \log \frac{(1+w)(1-w^k)}{(1-w)(1+w^k)} + \log(1-w)(1+w) - \log(1-w^k)(1+w^k) \right] \ . \end{aligned} \tag{6}$$

Now, setting the derivatives with respect to w to zero:

$$\frac{\partial F}{\partial w} = \eta g^k + \frac{1}{2\beta_k} \log \frac{(1+w)(1-w^k)}{(1-w)(1+w^k)} - y + z = 0 \ . \tag{7}$$

From complementary slackness conditions,

$$\begin{aligned} y(-w - 1) = 0 \ , \qquad \text{since} \quad w > -1 \ \Rightarrow \ y = 0 \ , \\ z(w - 1) = 0 \ , \qquad \text{since} \quad w < 1 \ \Rightarrow \ z = 0 \ . \end{aligned} \tag{8}$$

Algorithm 1 MD-tanh

Require: $K, b, \{\eta^k\}, \rho > 1, \mathcal{D}, L$
Ensure: $\mathbf{w}^* \in \mathcal{Q}^m$
$\mathbf{w}^0 \in \mathbb{R}^{\mathrm{m}}, \quad \beta_0 \leftarrow 1$ ▷ Initialization
$\mathbf{w}^0 \leftarrow \tanh(\beta_0 \mathbf{w}^0)$ ▷ Projection
for $k \leftarrow 0, \ldots, K$ **do**
  $\mathcal{D}^b = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^b \sim \mathcal{D}$ ▷ Sample a mini-batch
  $\mathbf{g}^k \leftarrow \nabla_{\mathbf{w}} L(\mathbf{w}; \mathcal{D}^b)\big|_{\mathbf{w}=\mathbf{w}^k}$ ▷ Gradient w.r.t. $\mathbf{w}$ at $\mathbf{w}^k$ (Adam based gradients)
  for $j \leftarrow 1, \ldots, m$ **do**
    $w_j^{k+1} \leftarrow \frac{\frac{1+w_j^k}{1-w_j^k} \exp(-2\beta_k \eta^k g_j^k) - 1}{\frac{1+w_j^k}{1-w_j^k} \exp(-2\beta_k \eta^k g_j^k) + 1}$ ▷ MD update
  end for
  $\beta_{k+1} \leftarrow \rho \beta_k$ ▷ Increase β
end for
$\mathbf{w}^* \leftarrow \mathrm{sign}(\tilde{\mathbf{w}}^K)$ ▷ Quantization

Algorithm 2 MD-tanh-S

Require: $K, b, \{\eta^k\}, \rho > 1, \mathcal{D}, L$
Ensure: $\mathbf{w}^* \in \mathcal{Q}^m$
$\tilde{\mathbf{w}}^0 \in \mathbb{R}^{\mathrm{m}}, \quad \beta_0 \leftarrow 1$ ▷ Initialization
for $k \leftarrow 0, \ldots, K$ **do**
  $\mathbf{w}^k \leftarrow \tanh(\beta_k \tilde{\mathbf{w}}^k)$ ▷ Projection
  $\mathcal{D}^b = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^b \sim \mathcal{D}$ ▷ Sample a mini-batch
  $\mathbf{g}^k \leftarrow \nabla_{\mathbf{w}} L(\mathbf{w}; \mathcal{D}^b)\big|_{\mathbf{w}=\mathbf{w}^k}$ ▷ Gradient w.r.t. $\mathbf{w}$ at $\mathbf{w}^k$ (Adam based gradients)
  $\tilde{\mathbf{w}}^{k+1} \leftarrow \tilde{\mathbf{w}}^k - \eta^k \mathbf{g}^k$ ▷ Gradient descent on $\tilde{\mathbf{w}}$
  $\beta_{k+1} \leftarrow \rho \beta_k$ ▷ Increase β
end for
$\mathbf{w}^* \leftarrow \mathrm{sign}(\tilde{\mathbf{w}}^K)$ ▷ Quantization

Therefore, Eq. (7) now simplifies to:

$$\begin{aligned} \frac{\partial F}{\partial w} &= \eta g^k + \frac{1}{2\beta_k} \log \frac{(1+w)(1-w^k)}{(1-w)(1+w^k)} = 0 \ , \\ \log \frac{(1+w)(1-w^k)}{(1-w)(1+w^k)} &= -2\beta_k \eta g^k \ , \\ \frac{1+w}{1-w} &= \frac{1+w^k}{1-w^k} \exp(-2\beta_k \eta g^k) \ , \\ w &= \frac{\frac{1+w^k}{1-w^k} \exp(-2\beta_k \eta g^k) - 1}{\frac{1+w^k}{1-w^k} \exp(-2\beta_k \eta g^k) + 1} \ . \end{aligned} \tag{9}$$

A similar derivation can also be performed for the sigmoid function, where $\bar{\mathcal{C}} = \mathcal{X} = [0, 1]$. Note that the sign function has been used for binary quantization in Courbariaux et al. (2015) and tanh can be used as a soft version of sign function as pointed out by Zhang et al. (2015). Mirror map corresponding to tanh is used for online linear optimization in Bubeck et al. (2012) but here we use it for NN quantization. The pseudocodes of original (MD-tanh) and numerically stable versions (MD-tanh-S) for tanh are presented in Algorithms 1 and 2 respectively.

Example 2 (**u**-space, multi-label, softmax)**.** Now we consider the softmax projection used in Proximal Mean-Field (PMF) (Ajanthan et al. (2019)) to optimize in the lifted probability space. In this case, the projection is defined as $P_{\beta_k}(\tilde{\mathbf{u}}) := \mathrm{softmax}(\beta_k \tilde{\mathbf{u}})$ where $P_{\beta_k} : \mathbb{R}^{\mathrm{d}} \to \mathcal{C}$ with $\bar{\mathcal{C}} = \mathcal{X} = \Delta$. Here Δ is the $(d-1)$-dimensional probability simplex and $|\mathcal{Q}| = d$. Note that the softmax projection is not invertible as it is a many-to-one mapping. In particular, it is invariant to translation, *i.e.*,

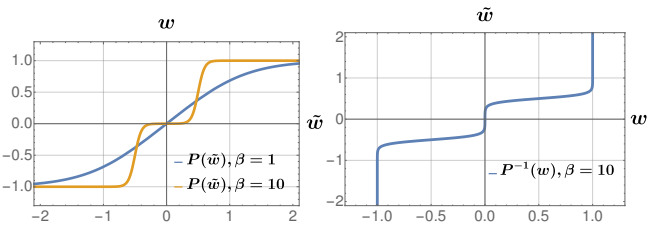

Figure 1: *Plots of shifted* tanh *projections, and their inverses corresponding to the* tanh *projection. Note that, the inverse is monotonically increasing. Moreover, when $\beta_k \to \infty$, the projections approaches their respective hard versions.*

$$\mathbf{u} = \text{softmax}(\tilde{\mathbf{u}} + c\mathbf{1}) = \text{softmax}(\tilde{\mathbf{u}}) \ ,$$
$$\text{where} \quad u_\lambda = \frac{\exp(\tilde{u}_\lambda)}{\sum_{\mu \in \mathcal{Q}} \exp(\tilde{u}_\mu)} \ ,$$

for any scalar $c \in \mathbb{R}$ ($\mathbf{1}$ denotes a vector of all ones). Therefore, the softmax projection does not satisfy Theorem 1 in the main paper. However, one could obtain a solution of the inverse of softmax as follows: given $\mathbf{u} \in \Delta$, find a unique point $\tilde{\mathbf{v}} = \tilde{\mathbf{u}} + c\mathbf{1}$, for a particular scalar c, such that $\mathbf{u} = \text{softmax}(\tilde{\mathbf{v}})$. Now, by choosing $c = -\log(\sum_{\mu=\mathcal{Q}} \exp(\tilde{u}_\mu))$, softmax can be written as:

$$\mathbf{u} = \text{softmax}(\tilde{\mathbf{v}}) \ , \qquad \text{where} \quad u_\lambda = \exp(\tilde{v}_\lambda) \ , \quad \forall\, \lambda \in \mathcal{Q} \ . \tag{10}$$

Now, the inverse of the projection P_{β_k} can be written as:

$$\tilde{\mathbf{v}} = P^{-1}_{\beta_k}(\mathbf{u}) = \frac{1}{\beta_k} \text{softmax}^{-1}(\mathbf{u}) \ , \qquad \text{where} \quad \tilde{v}_\lambda = \frac{1}{\beta_k} \log(u_\lambda) \ , \qquad \forall\, \lambda \ . \tag{11}$$

Indeed, log is a monotonically increasing function and from Theorem 1 in the main paper, by summing the integrals, the mirror map can be written as:

$$\Phi_{\beta_k}(\mathbf{u}) = \frac{1}{\beta_k} \left[\sum_\lambda u_\lambda \log(u_\lambda) - u_\lambda \right] = -\frac{1}{\beta_k} H(\mathbf{u}) - 1/\beta_k \ . \tag{12}$$

Here, $\sum_\lambda u_\lambda = 1$ as $\mathbf{u} \in \Delta$, and $H(\mathbf{u})$ is the entropy. Interestingly, as the mirror map in this case is the negative entropy (up to a constant), the MD update leads to the well-known Exponentiated Gradient Descent (EGD) (or Entropic Descent Algorithm (EDA)) (Beck and Teboulle (2003); Bubeck (2015)). Consequently, the update takes the following form:

$$u^{k+1}_\lambda = \frac{u^k_\lambda \exp(-\beta_k \eta g^k_\lambda)}{\sum_{\mu \in \mathcal{Q}} u^k_\mu \exp(-\beta_k \eta g^k_\mu)} \quad \forall\, \lambda \ . \tag{13}$$

The derivation follows the same approach as in the tanh case above. It is interesting to note that the MD variant of softmax is equivalent to the well-known EGD. Notice, the authors of PMF (Ajanthan et al. (2019)) hinted that PMF is related to EGD but here we have clearly showed that the MD variant of PMF under the above reparametrization (10) is exactly EGD.

Example 3 (**w**-space, multi-label, shifted tanh)**.** Note that, similar to softmax, we wish to extend the tanh projection beyond binary. The idea is to use a function that is an addition of multiple shifted tanh functions. To this end, as an example we consider ternary quantization, with $\mathcal{Q} = \{-1, 0, 1\}$ and define our shifted tanh projection $P_{\beta_k} : \mathbb{R} \to \mathcal{C}$ as:

$$w = P_{\beta_k}(\tilde{w}) = \frac{1}{2} \big[\tanh\left(\beta_k(\tilde{w} + 0.5)\right) + \tanh\left(\beta_k(\tilde{w} - 0.5)\right) \big] \ , \tag{14}$$

where $\beta_k \geq 1$ and $w = \bar{\mathcal{C}} = \mathcal{X} = [-1, 1]$. When $\beta_k \to \infty$, P_{β_k} approaches a stepwise function with inflection points at -0.5 and 0.5 (here, ± 0.5 is chosen heuristically), meaning w move towards one of the quantization levels in the set $\mathcal{Q}$. This behaviour together with its inverse is illustrated in Fig. 1. Now, one could potentially find the functional form of $P^{-1}_{\beta_k}$ and analytically derive the mirror map corresponding to this projection. Note that, while Theorem 1 in the main paper provides an analytical method to derive mirror maps, in some cases such as the above, the exact form of mirror map and the MD update might be nontrivial. In such cases, as shown in the paper, the MD update can be easily implemented by storing an additional set of auxiliary variables $\tilde{w}$.

A.2 Convergence Proof for MD with Adaptive Mirror Maps

Theorem 1. *Let $\mathcal{X} \subset \mathbb{R}^r$ be a convex compact set and $\mathcal{C} \subset \mathbb{R}^r$ be a convex open set with $\mathcal{X} \cap \mathcal{C} \neq \emptyset$ and $\mathcal{X} \subset \bar{\mathcal{C}}$. Let $\Phi : \mathcal{C} \to \mathbb{R}$ be a mirror map ρ-strongly convex on $\mathcal{X} \cap \mathcal{C}$ with respect to $\|\cdot\|$, $R^2 = \sup_{\mathbf{x} \in \mathcal{X} \cap \mathcal{C}} \Phi(\mathbf{x}) - \Phi(\mathbf{x}^0)$ where $\mathbf{x}^0 = \operatorname{argmin}_{\mathbf{x} \in \mathcal{X} \cap \mathcal{C}} \Phi(\mathbf{x})$ is the initialization, and $f : \mathcal{X} \to \mathbb{R}$ be a convex function and L-Lipschitz with respect to $\|\cdot\|$. Then MD with mirror map $\Phi_{\beta_k}(\mathbf{x}) = \Phi(\mathbf{x})/\beta_k$ with $1 \leq \beta_k \leq B$ and $\eta = \frac{R}{L}\sqrt{\frac{2\rho}{Bt}}$ satisfies*

$$f\left(\frac{1}{t}\sum_{k=0}^{t-1}\mathbf{x}^k\right) - f(\mathbf{x}^*) \leq RL\sqrt{\frac{2B}{\rho t}}\ , \tag{15}$$

where β_k is the annealing hyperparameter, $\eta > 0$ is the learning rate, t is the iteration index, and $\mathbf{x}^*$ is the optimal solution.

Proof. The proof is a slight modification to the proof of standard MD and we refer the reader to the proof of Theorem 4.2 of Bubeck (2015) for step by step derivation. We first discuss the intuition and then turn to the detailed proof. For the standard MD the bound is:

$$f\left(\frac{1}{t}\sum_{k=0}^{t-1}\mathbf{x}^k\right) - f(\mathbf{x}^*) \leq RL\sqrt{\frac{2}{\rho t}}\ , \tag{16}$$

with $\eta = \frac{R}{L}\sqrt{\frac{2\rho}{t}}$. Here, since $\beta_k \leq B$, the adaptive mirror map $\Phi_{\beta_k}(\mathbf{x}) = \Phi(\mathbf{x})/\beta_k$ is ρ/B-strongly convex for all k. Therefore, by simply replacing ρ with ρ/B the desired bound is obtained.

We now provide the step-by-step derivation for completeness. First note the MD update with the adaptive mirror map:

$$\begin{aligned}\nabla\Phi_{\beta_k}(\mathbf{y}^{k+1}) &= \nabla\Phi_{\beta_k}(\mathbf{x}^k) - \eta\mathbf{g}^k\ , \quad \text{where } \mathbf{g}^k \in \partial f(\mathbf{x}^k) \text{ and } \mathbf{y}^{k+1} \in \mathcal{C}\ , \\ \mathbf{g}^k &= (\nabla\Phi_{\beta_k}(\mathbf{x}^k) - \nabla\Phi_{\beta_k}(\mathbf{y}^{k+1}))/\eta, \quad \eta > 0\ .\end{aligned} \tag{17}$$

Now, let $\mathbf{x} \in \mathcal{X} \cap \mathcal{C}$. The claimed bound will be obtained by taking a limit $\mathbf{x} \to \mathbf{x}^*$.

$$\begin{aligned}f(\mathbf{x}^k) - f(\mathbf{x}) &\leq \langle \mathbf{g}^k, \mathbf{x}^k - \mathbf{x}\rangle\ , \quad f \text{ is convex}\ , \\ &= \langle \nabla\Phi_{\beta_k}(\mathbf{x}^k) - \nabla\Phi_{\beta_k}(\mathbf{y}^{k+1}), \mathbf{x}^k - \mathbf{x}\rangle/\eta\ , \quad \text{Eq. (17)}\ , \\ &= \left(D_{\Phi_{\beta_k}}(\mathbf{x}, \mathbf{x}^k) + D_{\Phi_{\beta_k}}(\mathbf{x}^k, \mathbf{y}^{k+1}) - D_{\Phi_{\beta_k}}(\mathbf{x}, \mathbf{y}^{k+1})\right)/\eta\ , \quad \text{Bregman div.}\ , \\ &\leq \left(D_{\Phi_{\beta_k}}(\mathbf{x}, \mathbf{x}^k) + D_{\Phi_{\beta_k}}(\mathbf{x}^k, \mathbf{y}^{k+1}) - D_{\Phi_{\beta_k}}(\mathbf{x}, \mathbf{x}^{k+1}) - D_{\Phi_{\beta_k}}(\mathbf{x}^{k+1}, \mathbf{y}^{k+1})\right)/\eta\ .\end{aligned} \tag{18}$$

The last line is due to the inequality $D_{\Phi_{\beta_k}}(\mathbf{x}, \mathbf{x}^{k+1}) + D_{\Phi_{\beta_k}}(\mathbf{x}^{k+1}, \mathbf{y}^{k+1}) \geq D_{\Phi_{\beta_k}}(\mathbf{x}, \mathbf{y}^{k+1})$, where $\mathbf{x}^{k+1} = \operatorname{argmin}_{\mathbf{x} \in \mathcal{X} \cap \mathcal{C}} D_{\Phi_{\beta_k}}(\mathbf{x}, \mathbf{y}^{k+1})$. Notice that,

$$\begin{aligned}\sum_{k=0}^{t-1} D_{\Phi_{\beta_k}}(\mathbf{x}, \mathbf{x}^k) - D_{\Phi_{\beta_k}}(\mathbf{x}, \mathbf{x}^{k+1}) &= \sum_{k=0}^{t-1}\left(D_\Phi(\mathbf{x}, \mathbf{x}^k) - D_\Phi(\mathbf{x}, \mathbf{x}^{k+1})\right)/\beta_k\ , \\ &= \beta_0^{-1}\left(D_\Phi(\mathbf{x}, \mathbf{x}^0) - D_\Phi(\mathbf{x}, \mathbf{x}^1)\right) + \beta_1^{-1}\left(D_\Phi(\mathbf{x}, \mathbf{x}^1) - D_\Phi(\mathbf{x}, \mathbf{x}^2)\right) + \ldots + \\ &\quad \beta_{t-1}^{-1}\left(D_\Phi(\mathbf{x}, \mathbf{x}^{t-1}) - D_\Phi(\mathbf{x}, \mathbf{x}^t)\right), \\ &= \beta_0^{-1}D_\Phi(\mathbf{x}, \mathbf{x}^0) + (\beta_1^{-1} - \beta_0^{-1})D_\Phi(\mathbf{x}, \mathbf{x}^1) + \ldots + (\beta_{t-1}^{-1} - \beta_{t-2}^{-1})D_\Phi(\mathbf{x}, \mathbf{x}^{t-1}) - \\ &\quad \beta_{t-1}^{-1}D_\Phi(\mathbf{x}, \mathbf{x}^t), \\ &\leq \beta_0^{-1}D_\Phi(\mathbf{x}, \mathbf{x}^0), \qquad D_\Phi(\mathbf{x}, \mathbf{z}) \geq 0,\ \forall\, \mathbf{x}, \mathbf{z} \in \mathcal{C} \text{ and } \beta_{k+1}^{-1} - \beta_k^{-1} < 0, \forall\, k \\ &\leq D_\Phi(\mathbf{x}, \mathbf{x}^0), \qquad \beta_0 \geq 1\end{aligned} \tag{19}$$

Now we bound the remaining term:

$$
\begin{aligned}
& D_{\Phi_{\beta_k}}(\mathbf{x}^k, \mathbf{y}^{k+1}) - D_{\Phi_{\beta_k}}(\mathbf{x}^{k+1}, \mathbf{y}^{k+1}) \qquad (20)\\
& = \Phi_{\beta_k}(\mathbf{x}^k) - \Phi_{\beta_k}(\mathbf{x}^{k+1}) - \langle \nabla\Phi_{\beta_k}(\mathbf{y}^{k+1}), \mathbf{x}^k - \mathbf{x}^{k+1}\rangle \ , \quad \text{Bregman divergence def.} \ , \\
& \leq \langle \nabla\Phi_{\beta_k}(\mathbf{x}^k) - \nabla\Phi_{\beta_k}(\mathbf{y}^{k+1}), \mathbf{x}^k - \mathbf{x}^{k+1}\rangle - \frac{\rho}{2\beta_k}\left\|\mathbf{x}^k - \mathbf{x}^{k+1}\right\|^2 \ , \quad \Phi \text{ is } \rho\text{-strongly convex} \ , \\
& = \langle \eta\mathbf{g}^k, \mathbf{x}^k - \mathbf{x}^{k+1}\rangle - \frac{\rho}{2\beta_k}\left\|\mathbf{x}^k - \mathbf{x}^{k+1}\right\|^2 \ , \quad \text{Eq. (17)} \ , \\
& \leq \eta L(\mathbf{x}^k - \mathbf{x}^{k+1}) - \frac{\rho}{2\beta_k}\left\|\mathbf{x}^k - \mathbf{x}^{k+1}\right\|^2 \ , \quad f \text{ is } L\text{-Lipschitz} \ , \\
& \leq \frac{(\eta L)^2\beta_k}{2\rho} \ , \qquad az - bz^2 \leq a^2/(4b), \ \forall\, z \in \mathbb{R} \ , \\
& \leq \frac{(\eta L)^2 B}{2\rho} \ , \qquad \beta_k \leq B \ .
\end{aligned}
$$

Putting Eqs. (19) and (20) in Eq. (18),

$$
\begin{aligned}
& \frac{1}{t}\sum_{k=0}^{t-1}\big(f(\mathbf{x}^k) - f(\mathbf{x})\big) \leq \frac{D_\Phi(\mathbf{x}, \mathbf{x}^0)}{\eta t} + \frac{\eta B L^2}{2\rho} \ , \qquad (21)\\
& f\left(\frac{1}{t}\sum_{k=0}^{t-1}\mathbf{x}^k\right) - f(\mathbf{x}) \leq \frac{R^2}{\eta t} + \frac{\eta B L^2}{2\rho} \ , \quad \text{Jensen inequality, defs. of } \mathbf{x}^0 \text{ and } R \ , \\
& \qquad\qquad = RL\sqrt{\frac{2B}{\rho t}} \ , \quad \text{Substituting } \eta = \tfrac{R}{L}\sqrt{\tfrac{2\rho}{Bt}} \ .
\end{aligned}
$$

Note the additional multiplication by $\sqrt{B}$ compared to the standard MD bound. However, the convergence rate is still $\mathcal{O}(1/\sqrt{t})$. □

A.3 Proof for Epsilon Convergence to a Discrete Solution via Annealing

Proposition 1. For a given $B > 0$ and $0 < \epsilon < 1$, there exists a $\gamma > 0$ such that if $|\tilde{x}| \geq \gamma$ then $1 - |\tanh(B\tilde{x})| < \epsilon$. Here $|\cdot|$ denotes the absolute value and $\gamma > \tanh^{-1}(1-\epsilon)/B$.

Proof. For a given B and ϵ, we derive a condition on $|\tilde{x}|$ for the inequality to be satisfied.

$$
\begin{aligned}
1 - |\tanh(B\tilde{x})| &< \epsilon \ , && (22)\\
|\tanh(B\tilde{x})| &> 1 - \epsilon \ , && (23)\\
\tanh(B|\tilde{x}|) &> 1 - \epsilon \ , \quad |\tanh(\tilde{x})| = \tanh(|\tilde{x}|) && (24)\\
|\tilde{x}| &> \tanh^{-1}(1-\epsilon)/B \ . \quad \tanh \text{ is monotone} && (25)
\end{aligned}
$$

Therefore for any $\gamma > \tanh^{-1}(1-\epsilon)/B$, the above inequality is satisfied. □

B Additional Experiments

We first give training curves of all compared methods, provide ablation study of ImageNet experiments as well as ternary quantization results as a proof of concept. Later, we provide experimental details.

B.1 Convergence Analysis

The training curves for CIFAR-10 and CIFAR-100 datasets with ResNet-18 are shown in Fig. 2. Notice, after the initial exploration phase (due to low β) the validation accuracies of our MD-tanh-S increase sharply while this steep rise is not observed in regularization methods such as PQ. The training behaviour for both our stable MD-variants (softmax and tanh) is quite similar.

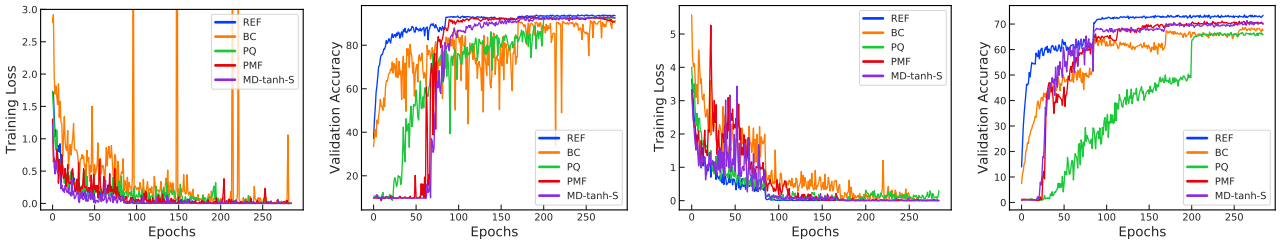

Figure 2: *Training curves for binarization for CIFAR-10 (first two columns) and CIFAR-100 (last two columns) with ResNet-18. Compared to BinaryConnect (*BC*), our* MD-tanh-S *and* PMF *are less noisy and after the initial exploration phase (up to* 60 *in CIFAR-10 and* 25 *epochs CIFAR-100), the validation accuracies rise sharply and closely resembles the floating point network afterwards. This steep increase is not observed in regularization methods such as ProxQuant (*PQ*).*

	Pretrained	Conv1 and FC	Bias	BN	Layerwise Scaling	Accuracy
MD-tanh-S	✓	Float	Float	Float	✗	**66.78/87.01**
	✗	Float	Float	Float	✗	65.92/86.29
	✓	Binary	Float	Float	✗	60.39/82.77
	✗	Binary	Float	Float	✗	59.92/82.42
	✓	Binary	Binary	Float	✗	56.60/79.79
	✗	Binary	Binary	Float	✗	56.67/79.66
	✓	Binary	Float	Float	✓	61.91/83.87
	✓	Binary	Binary	Float	✓	56.05/79.69

Table 1: *Ablation study on ImageNet with ResNet-18 for weights binarization using* MD-tanh-S*. While the best performance is obtained for the case where Conv1,* FC *and biases are not quantized,* MD-tanh-S *obtains good performance even when fully-quantized regardless of either using a pretrained network or training from scratch.*

B.2 ImageNet Ablation Study

We provide an ablation study for various experimental settings for weights binarization on ImageNet dataset using ResNet-18 architecture in Table 1. We perform experiments for both training for scratch and pretrained networks with variation in binarization of first convolution layer, fully connected layer and biases. Note that the performance degradation of our binary networks is minimal on all layers binarized network except biases using simple layerwise scaling as mentioned in McDonnell (2018). Contrary to the standard setup of binarized network training for ImageNet, where first and last layers are kept floating point, our MD-tanh-S method achieves good performance even on the fully-quantized network irrespective of either using a pretrained network or network trained from scratch.

	Algorithm	Space	CIFAR-10 VGG-16	CIFAR-10 ResNet-18	CIFAR-100 VGG-16	CIFAR-100 ResNet-18	TinyImageNet ResNet-18
	REF (float)	**w**	93.33	94.84	71.50	76.31	58.35
	PQ	**w**	83.32	90.50	32.16	59.18	41.46
	PQ*	**w**	**92.20**	**93.85**	57.64	70.98	45.72
	GD-tanh	**w**	91.21	93.20	53.88	69.48	50.65
Ours	MD-softmax-S	**u**	91.69	93.30	65.11	**72.01**	52.21
Ours	MD-tanh-S	**w**	91.70	93.42	**66.15**	71.29	**52.69**

Table 2: *Classification accuracies on the test set for ternary quantization.* PQ* *denotes performance with fully-connected layers, first convolution layer, and shortcut layers in floating point whereas* PQ *represent results with all layers quantized. Also,* PQ* *optimize for the quantization levels as well (different for each layer), in contrast we fix it to* $\mathcal{Q} = \{-1, 0, 1\}$. GD-tanh *denotes results without using* STE *and actually calculating the gradient through the projection.*

Dataset	Image	# class	Train / Val.	b	K
CIFAR-10	32×32	10	45k / 5k	128	100k
CIFAR-100	32×32	100	45k / 5k	128	100k
TinyImageNet	64×64	200	100k / 10k	128	100k
ImageNet	224×224	1000	1.2M / 50k	2048/256	90/55

Table 3: *Experiment setup. Here, b is the batch size and K is the total number of iterations for all datasets except ImageNet where K indicates number of epochs for training from scratch and pretrained network respectively. For ImageNet, b represents batch size for training from scratch and pretrained networks respectively.*

Hyperparameters	Fine-tuning grid
learning_rate	$[0.1, 0.01, 0.001, 0.0001]$
lr_scale	$[0.1, 0.2, 0.3, 0.5]$
beta_scale	$[1.01, 1.02, 1.05, 1.1, 1.2]$
beta_scale_interval	$[100, 200, 500, 1000, 2000]$

Table 4: *The hyperparameter search space for all the experiments. Chosen parameters are given in Tables 5, 6 and 7.*

B.3 Ternary Quantization Results

As a proof of concept for our shifted tanh projection (refer Example 3), we also show results for ternary quantization with quantization levels $\mathcal{Q} = \{-1, 0, 1\}$ in Table 2. Note that the performance improvement of our ternary networks compared to their respective binary networks is marginal as only 0 is included as the 3$^{\text{rd}}$ quantization level. In contrast to us, the baseline method PQ (Bai et al. (2019)) optimizes for the quantization levels (differently for each layer) as well in an alternating optimization regime rather than fixing it to $\mathcal{Q} = \{-1, 0, 1\}$. Also, PQ does ternarize the first convolution layer, fully-connected layers and the shortcut layers. We crossvalidate hyperparameters for both the original PQ setup and the equivalent setting of our MD-variants where we optimize all the weights and denote them as PQ* and PQ respectively.

Our MD-tanh variant performs on par or sometimes even better in comparison to tanh projection results where gradient is calculated through the projection instead of performing MD. This again empirically validates the hypothesis that MD yields in good approximation for the task of network quantization. The better performance of PQ in their original quantization setup, compared to our approach in CIFAR-10 can be accounted to their non-quantized layers and different quantization levels. We believe similar explorations are possible with our MD framework as well.

B.4 Experimental Details

As mentioned in the main paper the experimental protocol is similar to Ajanthan et al. (2019). To this end, the details of the datasets and their corresponding experiment setups are given in Table 3. For CIFAR-10/100 and TinyImageNet, VGG-16 (Simonyan and Zisserman (2015)), ResNet-18 (He et al. (2016)) and MobileNetV2 (Sandler et al. (2018)) architectures adapted for CIFAR dataset are used. In particular, for CIFAR experiments, similar to Lee et al. (2019), the size of the fully-connected (FC) layers of VGG-16 is set to 512 and no dropout layers are employed. For TinyImageNet, the stride of the first convolutional layer of ResNet-18 is set to 2 to handle the image size (Huang et al. (2017)). In all the models, batch normalization (Ioffe and Szegedy (2015)) (with no learnable parameters) and ReLU nonlinearity are used. Only for the floating point networks (*i.e.*, REF), we keep the learnable parameters for batch norm. Standard data augmentation (*i.e.*, random crop and horizontal flip) is used.

For both of our MD variants, hyperparameters such as the learning rate, learning rate scale, annealing hyperparameter β and its schedule are crossvalidated from the range reported in Table 4 and the chosen parameters are given in the Table 5, Table 6 and Table 7. To generate the plots, we used the publicly available codes of BC[1], PQ[2]

[1] `https://github.com/itayhubara/BinaryNet.pytorch`
[2] `https://github.com/allenbai01/ProxQuant`

	CIFAR-10 with ResNet-18							
	MD-softmax	MD-tanh	MD-softmax-S	MD-tanh-S	PMF*	GD-tanh	BC	PQ
learning_rate	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.01
lr_scale	0.2	0.3	0.3	0.3	0.3	0.3	0.3	0.5
beta_scale	1.02	1.01	1.02	1.02	1.1	1.1	-	0.0001
beta_scale_interval	200	100	200	200	1000	1000	-	-
	CIFAR-100 with ResNet-18							
	MD-softmax	MD-tanh	MD-softmax-S	MD-tanh-S	PMF*	GD-tanh	BC	PQ
learning_rate	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.1
lr_scale	0.2	0.3	0.2	0.2	0.3	0.5	0.2	-
beta_scale	1.05	1.05	1.1	1.2	1.01	1.01	-	0.001
beta_scale_interval	500	500	200	500	100	100	-	-
	CIFAR-10 with VGG-16							
	MD-softmax	MD-tanh	MD-softmax-S	MD-tanh-S	PMF*	GD-tanh	BC	PQ
learning_rate	0.01	0.001	0.001	0.001	0.001	0.001	0.0001	0.01
lr_scale	0.2	0.3	0.3	0.2	0.5	0.3	0.3	0.5
beta_scale	1.05	1.1	1.2	1.2	1.05	1.1	-	0.0001
beta_scale_interval	500	1000	2000	2000	500	1000	-	-
	CIFAR-100 with VGG-16							
	MD-softmax	MD-tanh	MD-softmax-S	MD-tanh-S	PMF*	GD-tanh	BC	PQ
learning_rate	0.001	0.001	0.0001	0.001	0.0001	0.001	0.0001	0.01
lr_scale	0.3	0.3	0.2	0.5	0.5	0.5	0.2	0.5
beta_scale	1.01	1.05	1.2	1.05	1.02	1.1	-	0.0001
beta_scale_interval	100	500	500	500	200	1000	-	-
	TinyImageNet with ResNet-18							
	MD-softmax	MD-tanh	MD-softmax-S	MD-tanh-S	PMF*	GD-tanh	BC	PQ
learning_rate	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.01
lr_scale	0.2	0.5	0.1	0.1	0.5	0.5	0.5	-
beta_scale	1.02	1.2	1.02	1.2	1.01	1.01	-	0.0001
beta_scale_interval	200	2000	100	500	100	100	-	-

Table 5: *Hyperparameter settings used for the binary quantization experiments. Here, the learning rate is multiplied by* lr_scale *after every 30k iterations and annealing hyperparameter* (β) *is multiplied by* beta_scale *after every* beta_scale_interval *iterations. We use Adam optimizer with zero weight decay. For* PQ*,* beta_scale *denotes regularization rate.*

and PMF[3].

All methods are trained from a random initialization and the model with the best validation accuracy is chosen for each method. Note that, in MD, even though we use an increasing schedule for β to enforce a discrete solution, the chosen network may not be fully-quantized (as the best model could be obtained in an early stage of training). Therefore, simple argmax rounding is applied to ensure that the network is fully-quantized.

B.5 ImageNet

We use the standard ResNet-18 architecture for ImageNet experiments where we train for 90 epochs and 55 epochs for training from scratch and pretrained network respectively. We perform all ImageNet experiments using NVIDIA DGX-1 machine with 8 Tesla V-100 GPUs for training from scratch and single Tesla V-100 GPU for training from a pretrained network. We provide detailed hyperparameter setup used for our experiments in Table 8. Similar to experiments on the other datasets, to enforce a discrete solution simple rounding based on sign operation is applied to ensure that the final network is fully-quantized. The final accuracy is reported based on the sign operation based discrete model obtained at the end of the final epoch.

[3] `https://github.com/tajanthan/pmf`

	CIFAR-10 with ResNet-18					
	REF (float)	MD-softmax-S	MD-tanh-S	GD-tanh	PQ	PQ*
learning_rate	0.1	0.001	0.01	0.01	0.01	0.01
lr_scale	0.3	0.3	0.2	0.5	0.3	-
beta_scale	-	1.05	1.2	1.02	0.0001	0.0001
beta_scale_interval	-	500	1000	500	-	-
weight_decay	0.0001	0	0	0	0	0.0001
	CIFAR-100 with ResNet-18					
	REF (float)	MD-softmax-S	MD-tanh-S	GD-tanh	PQ	PQ*
learning_rate	0.1	0.001	0.001	0.01	0.01	0.001
lr_scale	0.1	0.1	0.5	0.5	0.2	-
beta_scale	-	1.1	1.1	1.02	0.0001	0.0001
beta_scale_interval	-	100	500	1000	-	-
weight_decay	0.0001	0	0	0	0	0.0001
	CIFAR-10 with VGG-16					
	REF (float)	MD-softmax-S	MD-tanh-S	GD-tanh	PQ	PQ*
learning_rate	0.1	0.001	0.01	0.01	0.01	0.1
lr_scale	0.2	0.3	0.3	0.3	-	-
beta_scale	-	1.05	1.1	1.01	1e-07	0.0001
beta_scale_interval	-	500	1000	500	-	-
weight_decay	0.0001	0	0	0	0	0.0001
	CIFAR-100 with VGG-16					
	REF (float)	MD-softmax-S	MD-tanh-S	GD-tanh	PQ	PQ*
learning_rate	0.1	0.0001	0.001	0.01	0.01	0.0001
lr_scale	0.2	0.3	0.5	0.2	-	-
beta_scale	-	1.05	1.1	1.05	0.0001	0.0001
beta_scale_interval	-	100	500	2000	-	-
weight_decay	0.0001	0	0	0	0	0.0001
	TinyImageNet with ResNet-18					
	REF (float)	MD-softmax-S	MD-tanh-S	GD-tanh	PQ	PQ*
learning_rate	0.1	0.001	0.01	0.01	0.01	0.01
lr_scale	0.1	0.1	0.1	0.5	-	-
beta_scale	-	1.2	1.2	1.05	0.01	0.0001
beta_scale_interval	-	500	2000	2000	-	-
weight_decay	0.0001	0	0	0	0	0.0001

Table 6: *Hyperparameter settings used for the ternary quantization experiments. Here, the learning rate is multiplied by* lr_scale *after every 30k iterations and annealing hyperparameter* (β) *is multiplied by* beta_scale *after every* beta_scale_interval *iterations. We use Adam optimizer except for* REF *for which* SGD *with momentum* 0.9 *is used. For* PQ*,* beta_scale *denotes regularization rate.*

	CIFAR-10 with MobileNet-V2			
	REF (float)	BC	MD-softmax-S	MD-tanh-S
learning_rate	0.01	0.001	0.01	0.01
lr_scale	0.5	0.5	0.3	0.2
beta_scale	-	-	1.1	1.2
beta_scale_interval	-	-	1000	2000
weight_decay	0.0001	0	0	0
	CIFAR-100 with MobileNet-V2			
	REF (float)	BC	MD-softmax-S	MD-tanh-S
learning_rate	0.01	0.001	0.01	0.01
lr_scale	0.5	0.2	0.1	0.1
beta_scale	-	-	1.1	1.02
beta_scale_interval	-	-	500	100
weight_decay	0.0001	0	0	0

Table 7: *Hyperparameter settings used for the binary quantization experiments. Here, the learning rate is multiplied by* lr_scale *after every 30k iterations and annealing hyperparameter* (β) *is multiplied by* beta_scale *after every* beta_scale_interval *iterations. We use Adam optimizer except for* REF *for which* SGD *with momentum* 0.9 *is used.*

	ImageNet with ResNet-18		
	REF (float)	MD-tanh-S*	MD-tanh-S
base_learning_rate	2.048	2.048	0.0768
warmup_epochs	8	8	0
beta_scale	-	1.02	1.02
beta_scale_interval (iterations)	-	62	275
batch_size	2048	2048	256
weight_decay	3.0517e-05	3.0517e-05	3.0517e-05

Table 8: *Hyperparameter settings used for the binary quantization experiments on ImageNet dataset using ResNet-18 architecture. Here* MD-tanh-S* *is trained from scratch while* MD-tanh-S *is finetuned on the pretrained network. We use* SGD *optimizer with momentum* 0.875 *and cosine learning rate scheduler for all experiments. For all the experiments, weight decay in batchnorm layers are off. Similar to Goyal et al. (2017), for experiments with larger batch size we use gradual warmup where learning rate is linearly scaled from small learning rate to the base learning rate. Also, note that training schedule is fixed based on above hyperparameters for ablation study on ImageNet dataset.*

References

Ajanthan, T., Dokania, P. K., Hartley, R., and Torr, P. H. (2019). Proximal mean-field for neural network quantization. *ICCV*.

Bai, Y., Wang, Y.-X., and Liberty, E. (2019). Proxquant: Quantized neural networks via proximal operators. *ICLR*.

Beck, A. and Teboulle, M. (2003). Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*.

Bubeck, S. (2015). Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*.

Bubeck, S., Cesa-Bianchi, N., and Kakade, S. M. (2012). Towards minimax policies for online linear optimization with bandit feedback. *Conference on Learning Theory*.

Courbariaux, M., Bengio, Y., and David, J.-P. (2015). Binaryconnect: Training deep neural networks with binary weights during propagations. *NeurIPS*.

Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., and He, K. (2017). Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *CVPR*.

Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., and Weinberger, K. Q. (2017). Snapshot ensembles: Train 1, get m for free. *ICLR*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*.

Lee, N., Ajanthan, T., and Torr, P. H. S. (2019). SNIP: Single-shot network pruning based on connection sensitivity. *ICLR*.

McDonnell, M. D. (2018). Training wide residual networks for deployment using a single bit for each weight. *ICLR*.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). Mobilenetv2: inverted residuals and linear bottlenecks. *CVPR*.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *ICLR*.

Zhang, R., Lin, L., Zhang, R., Zuo, W., and Zhang, L. (2015). Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. *TIP*.