

# Mirror Descent View for Neural Network Quantization

Thalaiyasingam Ajanthan<sup>\*1</sup>, Kartik Gupta<sup>\*1,3</sup>, Philip H. S. Torr<sup>2</sup>, Richard Hartley<sup>1</sup>, and Puneet K. Dokania<sup>2</sup>

<sup>1</sup> Australian National University

<sup>2</sup> University of Oxford

<sup>3</sup> Data61, CSIRO

**Abstract.** Quantizing large Neural Networks (NN) while maintaining the performance is highly desirable for resource-limited devices due to reduced memory and time complexity. It is usually formulated as a constrained optimization problem and optimized via a modified version of gradient descent. In this work, by interpreting the continuous parameters (unconstrained) as the dual of the quantized ones, we introduce a Mirror Descent (MD) framework [7] for NN quantization. Specifically, we provide conditions on the projections (*i.e.*, mapping from continuous to quantized ones) which would enable us to derive valid mirror maps and in turn the respective MD updates. Furthermore, we present a numerically stable implementation of MD that requires storing an additional set of auxiliary variables (unconstrained), and show that it is strikingly analogous to the Straight Through Estimator (STE) based method which is typically viewed as a “trick” to avoid vanishing gradients issue. Our experiments on CIFAR-10/100, TinyImageNet, and ImageNet classification datasets with VGG-16, ResNet-18, and MobileNetV2 architectures show that our MD variants obtain quantized networks with state-of-the-art performance.

## 1 Introduction

Despite the success of deep neural networks in various domains, their excessive computational and memory requirements limit their practical usability for real-time applications or in resource-limited devices. Quantization is a prominent technique for network compression, where the objective is to learn a network while restricting the parameters to take values from a small discrete set. This leads to a dramatic reduction in memory (a factor of 32 for binary quantization) and inference time – as it enables specialized implementation using bit operations.

Neural Network (NN) quantization is usually formulated as a constrained optimization problem  $\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$ , where  $f(\cdot)$  denotes the loss function by abstracting out the dependency on the dataset and  $\mathcal{X} \subset \mathbb{R}^r$  denotes the set of all possible quantized solutions. Majority of the works in the literature [3,19,34] convert this into an unconstrained problem by introducing auxiliary variables

---

<sup>\*</sup> Equal contribution.

$(\tilde{\mathbf{x}})$  and optimize via (stochastic) gradient descent. Specifically, the objective and the update step take the following form:

$$\min_{\tilde{\mathbf{x}} \in \mathbb{R}^r} f(P(\tilde{\mathbf{x}})) , \quad \tilde{\mathbf{x}}^{k+1} = \tilde{\mathbf{x}}^k - \eta \nabla_{\tilde{\mathbf{x}}} f(P(\tilde{\mathbf{x}}))|_{\tilde{\mathbf{x}}=\tilde{\mathbf{x}}^k} , \quad (1)$$

where  $P : \mathbb{R}^r \rightarrow \mathcal{X}$  is a mapping from the unconstrained space to the quantized space (sometimes called projection) and  $\eta > 0$  is the learning rate. In cases where the mapping  $P$  is not differentiable, a suitable approximation is employed [19].

In this work, by noting that the well-known Mirror Descent (MD) algorithm, widely used for online convex optimization [7], provides a theoretical framework to perform gradient descent in the unconstrained space (dual space,  $\mathbb{R}^r$ ) with gradients computed in the quantized space (primal space,  $\mathcal{X}$ ), we introduce an MD framework for NN quantization. In essence, MD extends gradient descent to non-Euclidean spaces where Euclidean projection is replaced with a more general projection defined based on the associated distance metric. Briefly, the key ingredient of MD is a concept called *mirror map* which defines both the mapping between primal and dual spaces and the exact form of the projection. Specifically, in this work, by observing  $P$  in Eq. (1) as a mapping from dual space to the primal space, we analytically derive corresponding mirror maps under certain conditions on  $P$ . This enables us to derive different variants of the MD algorithm useful for NN quantization.

Note that, MD requires the constrained set to be convex, however, the quantization set is discrete. Therefore, as discussed later in Sec. 3, to ensure quantized solutions, we employ a monotonically increasing annealing hyperparameter similar to [3,4]. This translates into *time-varying mirror maps*, and for completeness, we theoretically analyze the convergence behaviour of MD in this case for the convex setting. Furthermore, as MD is often found to be numerically unstable [17], we discuss a numerically stable implementation of MD by storing an additional set of auxiliary variables. In fact, this update is strikingly analogous to the popular Straight Through Estimator (STE) based gradient method [4,19] which is typically viewed as a “trick” to avoid vanishing gradients issue but here we show that it is an implementation method for MD under certain conditions on the mapping  $P$ . We believe this connection sheds some light on the practical effectiveness of STE.

We evaluate the merits of our MD variants on CIFAR-10/100, TinyImageNet and ImageNet classification datasets with VGG-16, ResNet-18, and MobileNetV2 architectures. Our experiments show that the quantized networks obtained by the MD variants outperform comparable baselines and yield state-of-the-art performance where the accuracies are very close to the floating-point counterparts.

## 2 Preliminaries

Here we provide a brief background on the MD algorithm and NN quantization.

### 2.1 Mirror Descent

The Mirror Descent (MD) algorithm was first introduced in [27] and has extensively been studied in the convex optimization literature ever since. In this section we

provide a brief overview and refer the interested reader to Chapter 4 of [7]. In the context of MD, we consider a problem of the form:

$$\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) , \quad (2)$$

where  $f : \mathcal{X} \rightarrow \mathbb{R}$  is a convex function and  $\mathcal{X} \subset \mathbb{R}^r$  is a compact convex set. The main concept of MD is to extend gradient descent to a more general non-Euclidean space (Banach space<sup>4</sup>), thus overcoming the dependency of gradient descent on the Euclidean geometry. The motivation for this generalization is that one might be able to exploit the geometry of the space to optimize much more efficiently. One such example is the simplex constrained optimization where MD converges at a much faster rate than the standard Projected Gradient Descent (PGD).

To this end, since the gradients lie in the dual space, optimization is performed by first mapping the primal point  $\mathbf{x}^k \in \mathcal{B}$  (quantized space,  $\mathcal{X}$ ) to the dual space  $\mathcal{B}^*$  (unconstrained space,  $\mathbb{R}^r$ ), then performing gradient descent in the dual space, and finally mapping back the resulting point to the primal space  $\mathcal{B}$ . If the new point  $\mathbf{x}^{k+1}$  lie outside of the constraint set  $\mathcal{X} \subset \mathcal{B}$ , it is projected to the set  $\mathcal{X}$ . Both the primal/dual mapping and the projection are determined by the *mirror map*. Specifically, the gradient of the mirror map defines the mapping from primal to dual and the projection is done via the Bregman divergence of the mirror map. We first provide the definitions for mirror map and Bregman divergence and then turn to the MD updates.

**Definition 1** (Mirror map). Let  $\mathcal{C} \subset \mathbb{R}^r$  be a convex open set such that  $\mathcal{X} \subset \bar{\mathcal{C}}$  ( $\bar{\mathcal{C}}$  denotes the closure of set  $\mathcal{C}$ ) and  $\mathcal{X} \cap \mathcal{C} \neq \emptyset$ . Then,  $\Phi : \mathcal{C} \rightarrow \mathbb{R}$  is a mirror map if it satisfies:

1.  $\Phi$  is strictly convex and differentiable.
2.  $\nabla \Phi(\mathcal{C}) = \mathbb{R}^r$ , i.e.,  $\nabla \Phi$  takes all possible values in  $\mathbb{R}^r$ .
3.  $\lim_{\mathbf{x} \rightarrow \partial \mathcal{C}} \|\nabla \Phi(\mathbf{x})\| = \infty$  ( $\partial \mathcal{C}$  denotes the boundary of  $\mathcal{C}$ ), i.e.,  $\nabla \Phi$  diverges on the boundary of  $\mathcal{C}$ .

**Definition 2** (Bregman divergence). Let  $\Phi : \mathcal{C} \rightarrow \mathbb{R}$  be a continuously differentiable, strictly convex function defined on a convex set  $\mathcal{C}$ . The Bregman divergence associated with  $\Phi$  for points  $\mathbf{p}, \mathbf{q} \in \mathcal{C}$  is the difference between the value of  $\Phi$  at point  $\mathbf{p}$  and the value of the first-order Taylor expansion of  $\Phi$  around point  $\mathbf{q}$  evaluated at point  $\mathbf{p}$ , i.e.,

$$D_\Phi(\mathbf{p}, \mathbf{q}) = \Phi(\mathbf{p}) - \Phi(\mathbf{q}) - \langle \nabla \Phi(\mathbf{q}), \mathbf{p} - \mathbf{q} \rangle . \quad (3)$$

Notice,  $D_\Phi(\mathbf{p}, \mathbf{q}) \geq 0$  with  $D_\Phi(\mathbf{p}, \mathbf{p}) = 0$ , and  $D_\Phi(\mathbf{p}, \mathbf{q})$  is convex on  $\mathbf{p}$ .

Now we are ready to provide the mirror descent strategy based on the mirror map  $\Phi$ . Let  $\mathbf{x}^0 \in \arg\min_{\mathbf{x} \in \mathcal{X} \cap \mathcal{C}} \Phi(\mathbf{x})$  be the initial point. Then, for iteration  $k \geq 0$  and step size  $\eta > 0$ , the update of the MD algorithm can be written as:

$$\begin{aligned} \nabla \Phi(\mathbf{y}^{k+1}) &= \nabla \Phi(\mathbf{x}^k) - \eta \mathbf{g}^k , \quad \text{where } \mathbf{g}^k \in \partial f(\mathbf{x}^k) \text{ and } \mathbf{y}^{k+1} \in \mathcal{C} , \\ \mathbf{x}^{k+1} &= \arg\min_{\mathbf{x} \in \mathcal{X} \cap \mathcal{C}} D_\Phi(\mathbf{x}, \mathbf{y}^{k+1}) . \end{aligned} \quad (4)$$

<sup>4</sup> A Banach space is a complete normed vector space where the norm is not necessarily derived from an inner product.

Note that, in Eq. (4), the gradient  $\mathbf{g}^k$  is computed at  $\mathbf{x}^k \in \mathcal{X} \cap \mathcal{C}$  (solution space) but the gradient descent is performed in  $\mathbb{R}^r$  (unconstrained dual space). Moreover, by simple algebraic manipulation, it is easy to show that the above MD update (4) can be compactly written in a proximal form where the Bregman divergence of the mirror map becomes the proximal term [5]:

$$\mathbf{x}^{k+1} = \operatorname{argmin}_{\mathbf{x} \in \mathcal{X} \cap \mathcal{C}} \langle \eta \mathbf{g}^k, \mathbf{x} \rangle + D_\Phi(\mathbf{x}, \mathbf{x}^k) . \quad (5)$$

Note, if  $\Phi(\mathbf{x}) = \frac{1}{2} \|\mathbf{x}\|_2^2$ , then  $D_\Phi(\mathbf{x}, \mathbf{x}^k) = \frac{1}{2} \|\mathbf{x} - \mathbf{x}^k\|_2^2$ , which when plugged back to the above problem and optimized for  $\mathbf{x}$ , leads to exactly the same update rule as that of PGD. However, MD allows us to choose various forms of  $\Phi$  depending on the problem at hand.

## 2.2 Neural Network Quantization

Neural Network (NN) quantization amounts to training networks with parameters restricted to a small discrete set representing the quantization levels. Here we review two constrained optimization formulations for NN quantization: 1) directly constrain each parameter to be in the discrete set; and 2) optimize the probability of each parameter taking a label from the set of quantization levels.

**Parameter Space Formulation** Given a dataset  $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n$ , NN quantization can be written as:

$$\min_{\mathbf{w} \in \mathcal{Q}^m} L(\mathbf{w}; \mathcal{D}) := \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{w}; (\mathbf{x}_i, \mathbf{y}_i)) . \quad (6)$$

Here,  $\ell(\cdot)$  denotes the input-output mapping composed with a standard loss function (*e.g.*, cross-entropy loss),  $\mathbf{w}$  is the  $m$  dimensional parameter vector, and  $\mathcal{Q}$  with  $|\mathcal{Q}| = d$  is a predefined discrete set representing quantization levels (*e.g.*,  $\mathcal{Q} = \{-1, 1\}$  or  $\mathcal{Q} = \{-1, 0, 1\}$ ).

The approaches that directly optimize in the parameter space include BinaryConnect (BC) [11] and its variants [19, 29], where the constraint set is discrete. In contrast, recent approaches [4, 34] relax this constraint set to be its convex hull:

$$\operatorname{conv}(\mathcal{Q}^m) = [q_{\min}, q_{\max}]^m , \quad (7)$$

where  $q_{\min}$  and  $q_{\max}$  represent the minimum and maximum quantization levels, respectively. In this case, a quantized solution is obtained by gradually increasing an annealing hyperparameter.

**Lifted Probability Space Formulation** Another formulation is based on the Markov Random Field (MRF) perspective to NN quantization recently studied in [3]. It treats Eq. (6) as a *discrete labelling problem* and introduces indicator variables  $u_{j:\lambda} \in \{0, 1\}$  for each parameter  $w_j$  where  $j \in \{1, \dots, m\}$  such that  $u_{j:\lambda} = 1$  if and only if  $w_j = \lambda \in \mathcal{Q}$ . For convenience, by denoting the vector of quantization levels as  $\mathbf{q}$ , a parameter vector  $\mathbf{w} \in \mathcal{Q}^m$  can be written in a matrix vector product as:

$$\mathbf{w} = \mathbf{u} \mathbf{q} , \quad \text{where} \quad \mathbf{u} \in \mathcal{V}^m = \left\{ \mathbf{u} \left| \begin{array}{ll} \sum_{\lambda} u_{j:\lambda} = 1, & \forall j \\ u_{j:\lambda} \in \{0, 1\}, & \forall j, \lambda \end{array} \right. \right\} . \quad (8)$$

Here,  $\mathbf{u}$  is a  $m \times d$  matrix (*i.e.*, each row  $\mathbf{u}_j = \{u_{j:\lambda} \mid \lambda \in \mathcal{Q}\}$ ), and  $\mathbf{q}$  is a column vector of dimension  $d$ . Note that,  $\mathbf{u} \in \mathcal{V}^m$  is an overparametrized (*i.e.*, lifted) representation of  $\mathbf{w} \in \mathcal{Q}^m$ . Now, similar to the relaxation in the parameter space, one can relax the binary constraint in  $\mathcal{V}^m$  to form its convex hull:

$$\Delta^m = \text{conv}(\mathcal{V}^m) = \left\{ \mathbf{u} \mid \begin{array}{ll} \sum_{\lambda} u_{j:\lambda} = 1, & \forall j \\ u_{j:\lambda} \geq 0, & \forall j, \lambda \end{array} \right\}. \quad (9)$$

The set  $\Delta^m$  is in fact the Cartesian product of the standard  $(d-1)$ -probability simplexes embedded in  $\mathbb{R}^d$ . Therefore, for a feasible point  $\mathbf{u} \in \Delta^m$ , the vector  $\mathbf{u}_j$  for each  $j$  ( $j$ -th row of matrix  $\mathbf{u}$ ) belongs to the probability simplex  $\Delta$ . Hence, we can interpret the value  $u_{j:\lambda}$  as the probability of assigning the discrete label  $\lambda$  to the weight  $w_j$ . This relaxed optimization can then be written as:

$$\min_{\mathbf{u} \in \Delta^m} L(\mathbf{u}\mathbf{q}; \mathcal{D}) := \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{u}\mathbf{q}; (\mathbf{x}_i, \mathbf{y}_i)). \quad (10)$$

In fact, this can be interpreted as finding a probability distribution  $\mathbf{u} \in \Delta^m$  such that the cost  $L(\mathbf{u})$  is minimized. Note that, the relaxation of  $\mathbf{u}$  from  $\mathcal{V}^m$  to  $\Delta^m$  translates into relaxing  $\mathbf{w}$  from  $\mathcal{Q}^m$  to the convex region  $\text{conv}(\mathcal{Q}^m)$ . Even in this case, a discrete solution  $\mathbf{u} \in \mathcal{V}^m$  can be enforced via an annealing hyperparameter or using rounding schemes.

### 3 Mirror Descent Framework for Network Quantization

Before introducing the MD formulation, we first write NN quantization as a single objective unifying (6) and (10) as:

$$\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}), \quad (11)$$

where  $f(\cdot)$  denotes the loss function by abstracting out the dependency on the dataset  $\mathcal{D}$ , and  $\mathcal{X}$  denotes the constraint set ( $\mathcal{Q}^m$  or  $\mathcal{V}^m$  depending on the formulation). As discussed in Sec. 2.2, many recent NN quantization methods optimize over the convex hull of the constraint set. Following this, we consider the solution space  $\mathcal{X}$  in Eq. (11) to be convex and compact. To employ MD, we need to choose a mirror map (refer Definition 1) suitable for the problem at hand. In fact, as discussed in Sec. 2.1, mirror map is the core component of an MD algorithm which determines the effectiveness of the resulting MD updates. However, there is no straightforward approach to obtain a mirror map for a given constrained optimization problem, except in certain special cases.

To this end, we observe that the usual approach to optimize the above constrained problem is via a version of projected gradient descent, where the projection is the mapping from the unconstrained auxiliary variables (high-precision) to the quantized space  $\mathcal{X}$ . Now, noting the analogy between the purpose of the projection operator and the mirror maps in the MD formulation, we intend to derive the mirror map analogous to a given projection. Precisely, we prove that if the *projection is invertible and strictly monotone*, a valid mirror map can be derived from the projection itself. This does not necessarily extend

the theory of MD as finding a strictly monotone map is as hard as finding the mirror map itself. However, this derivation is valuable as it connects existing PGD type algorithms to their corresponding MD variants. For completeness, we state it as a theorem for the case  $\mathcal{X} \subset \mathbb{R}$  and the multidimensional case can be proved with an additional assumption that the vector field  $P^{-1}(\mathbf{x})$  is conservative.

**Theorem 1.** *Let  $\mathcal{X}$  be a compact convex set and  $P : \mathbb{R} \rightarrow \mathcal{C}$  be an invertible function where  $\mathcal{C} \subset \mathbb{R}$  is a convex open set such that  $\mathcal{X} = \bar{\mathcal{C}}$  ( $\bar{\mathcal{C}}$  denotes the closure of  $\mathcal{C}$ ). Now, if*

1.  *$P$  is strictly monotonically increasing.*
2.  *$\lim_{x \rightarrow \partial \mathcal{C}} \|P^{-1}(x)\| = \infty$  ( $\partial \mathcal{C}$  denotes the boundary of  $\mathcal{C}$ ).*

*Then,  $\Phi(x) = \int_{x_0}^x P^{-1}(y)dy$  is a valid mirror map.*

*Proof.* This can be proved by noting that  $\nabla \Phi(x) = P^{-1}(x)$ . Please refer to Appendix A.1.

The MD update based on the mirror map derived from a given projection is illustrated in Fig. 1. Note that, to employ MD to the problem (11), in theory, any mirror map satisfying Definition 1 whose domain (*i.e.*, its closure) is a superset of the constraint set  $\mathcal{X}$  can be chosen. The above theorem provides a method to derive a subset of all applicable mirror maps, where the closure of the domain of mirror maps is exactly equal to the constraint set  $\mathcal{X}$ .

We now give some example projections useful for NN quantization (tanh for **w**-space and softmax for **u**-space) and derive their corresponding mirror maps. Given mirror maps, the MD updates are straightforward based on Eq. (5). Even though we consider differentiable projections, Theorem 1 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  $[-1, 1]$ :

$$w = P(\tilde{w}) := \tanh(\beta \tilde{w}) = \frac{\exp(2\beta \tilde{w}) - 1}{\exp(2\beta \tilde{w}) + 1}, \quad (12)$$

where  $\beta > 0$  is the annealing hyperparameter and when  $\beta \rightarrow \infty$ , tanh approaches the step function. The inverse of the tanh is:

$$P^{-1}(w) = \frac{1}{\beta} \tanh^{-1}(w) = \frac{1}{2\beta} \log \frac{1+w}{1-w}. \quad (13)$$

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

$$\Phi(w) = \int P^{-1}(w)dw = \frac{1}{2\beta} [(1+w) \log(1+w) + (1-w) \log(1-w)]. \quad (14)$$

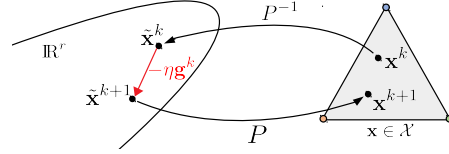


Fig. 1: MD formulation where mirror map is derived from the projection  $P$ . Note,  $\mathbf{g}^k$  is computed in the primal space ( $\mathcal{X}$ ) but it is directly used to update the auxiliary variables in the dual space.

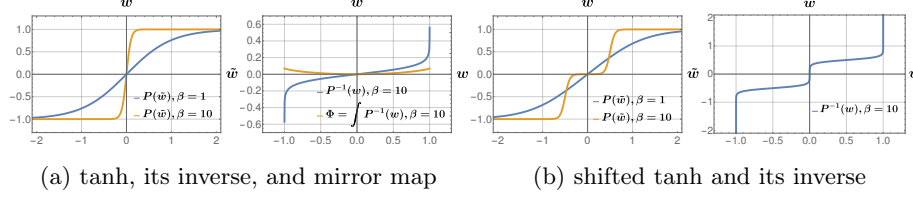


Fig. 2: Plots of tanh and shifted tanh projections, and their inverses corresponding to the tanh projection. Note that, the inverses are monotonically increasing and the mirror map is strictly convex. Moreover, when  $\beta \rightarrow \infty$ , the projections approaches their respective hard versions.

Here, the constant from the integration is ignored. It can be easily verified that  $\Phi(w)$  is in fact a valid mirror map. The projection, its inverse and the corresponding mirror map are illustrated in Fig. 2a. Consequently, the resulting MD update (5) takes the following form:

$$w^{k+1} = \underset{w \in (-1,1)}{\operatorname{argmin}} \langle \eta g^k, w \rangle + D_\Phi(w, w^k) = \frac{\frac{1+w^k}{1-w^k} \exp(-2\beta\eta g^k) - 1}{\frac{1+w^k}{1-w^k} \exp(-2\beta\eta g^k) + 1}. \quad (15)$$

The update formula is derived using the KKT conditions [6]. For the detailed derivation please refer to Appendix A.2. 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 [11] and tanh can be used as a soft version of sign function as pointed out by [35]. Mirror map corresponding to tanh is used for online linear optimization in [8] but here we use it for NN quantization. The pseudocode of the above mentioned approach (MD-tanh) is provided in Algorithm 1 in the Appendix.

**Example 2 (u-space, multi-label, softmax).** Now we consider the softmax projection used in Proximal Mean-Field (PMF) [3] to optimize in the lifted probability space. In this case, the projection is defined as  $P(\tilde{\mathbf{u}}) := \operatorname{softmax}(\beta \tilde{\mathbf{u}})$  where  $P : \mathbb{R}^d \rightarrow \mathcal{C}$  with  $\bar{\mathcal{C}} = \mathcal{X} = \Delta$ . Here  $\Delta$  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.*,

$$\mathbf{u} = \operatorname{softmax}(\tilde{\mathbf{u}} + c\mathbf{1}) = \operatorname{softmax}(\tilde{\mathbf{u}}), \quad \text{where } u_\lambda = \frac{\exp(\tilde{u}_\lambda)}{\sum_{\mu \in \mathcal{Q}} \exp(\tilde{u}_\mu)}, \quad (16)$$

for any scalar  $c \in \mathbb{R}$  ( $\mathbf{1}$  denotes a vector of all ones). Therefore, the softmax projection does not satisfy Theorem 1. 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  $\mathbf{u} = \operatorname{softmax}(\tilde{\mathbf{v}})$ . Now, by choosing  $c = -\log(\sum_{\mu \in \mathcal{Q}} \exp(\tilde{u}_\mu))$ , softmax can be written as:

$$\mathbf{u} = \operatorname{softmax}(\tilde{\mathbf{v}}), \quad \text{where } u_\lambda = \exp(\tilde{v}_\lambda), \quad \forall \lambda \in \mathcal{Q}. \quad (17)$$

Now, the inverse of the projection can be written as:

$$\tilde{\mathbf{v}} = P^{-1}(\mathbf{u}) = \frac{1}{\beta} \operatorname{softmax}^{-1}(\mathbf{u}), \quad \text{where } \tilde{v}_\lambda = \frac{1}{\beta} \log(u_\lambda), \quad \forall \lambda. \quad (18)$$

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

$$\Phi(\mathbf{u}) = \frac{1}{\beta} \left[ \sum_{\lambda} u_{\lambda} \log(u_{\lambda}) - u_{\lambda} \right] = -\frac{1}{\beta} H(\mathbf{u}) - 1/\beta . \quad (19)$$

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)) [5,7]. Consequently, the update takes the following form:

$$u_{\lambda}^{k+1} = \frac{u_{\lambda}^k \exp(-\beta \eta g_{\lambda}^k)}{\sum_{\mu \in \mathcal{Q}} u_{\mu}^k \exp(-\beta \eta g_{\mu}^k)} \quad \forall \lambda . \quad (20)$$

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 [3] hinted that PMF is related to EGD but here we have clearly showed that the MD variant of PMF under the above reparametrization (17) 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  $P : \mathbb{R} \rightarrow \mathcal{C}$  as:

$$w = P(\tilde{w}) = \frac{1}{2} \left[ \tanh(\beta(\tilde{w} + 0.5)) + \tanh(\beta(\tilde{w} - 0.5)) \right] , \quad (21)$$

where  $\beta \geq 1$  and  $w = \bar{\mathcal{C}} = \mathcal{X} = [-1, 1]$ . When  $\beta \rightarrow \infty$ ,  $P$  approaches a stepwise function with inflection points at  $-0.5$  and  $0.5$  (here,  $\pm 0.5$  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. 2b. Now, one could potentially find the functional form of  $P^{-1}$  and analytically derive the mirror map corresponding to this projection. Note that, while Theorem 1 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 will be shown subsequently, the MD update can be easily implemented by storing an additional set of auxiliary variables  $\tilde{w}$ .

### 3.1 Effect of Annealing and Convergence Analysis

Note that, to ensure a discrete solution, projection  $P$  is parametrized by a scalar  $\beta \geq 1$  and it is annealed throughout the optimization. This annealing hyperparameter translates into a time varying mirror map (refer Eqs. (14) and (19)) in our case. Such an adaptive mirror map gradually constrains the solution space  $\mathcal{X}$  to its boundary and in the limit enforces a quantized solution. The classical MD literature theoretically studied the convergence behaviour of MD for the convex



setting, however, to the best of our knowledge adaptive mirror maps have not been considered before. To this end, we now prove that, in the convex setting, if the annealing hyperparameter  $\beta$  is bounded, then MD with an adaptive mirror map converges to the optimum at the same rate of  $\mathcal{O}(1/\sqrt{t})$  as the standard MD.

**Theorem 2.** *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  $\mathcal{X} \subset \bar{\mathcal{C}}$  ( $\bar{\mathcal{C}}$  denotes the closure of  $\mathcal{C}$ ). Let  $\Phi : \mathcal{C} \rightarrow \mathbb{R}$  be a mirror map  $\rho$ -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  $f : \mathcal{X} \rightarrow \mathbb{R}$  be a convex function and  $L$ -Lipschitz with respect to  $\|\cdot\|$ . Then MD with mirror map  $\Phi_\beta(\mathbf{x}) = \Phi(\mathbf{x})/\beta$  with  $1 \leq \beta \leq B$  and  $\eta = \frac{R}{L} \sqrt{\frac{2\rho}{Bt}}$  satisfies*

$$f\left(\frac{1}{t} \sum_{k=0}^{t-1} \mathbf{x}^k\right) - f(\mathbf{x}^*) \leq RL \sqrt{\frac{2B}{\rho t}}, \quad (22)$$

where  $\beta$  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 noting that, effectively  $\Phi_\beta$  is  $\rho/B$ -strongly convex. Please refer to Appendix A.3.

According to the theorem, in our implementation,  $\beta$  is capped at an arbitrarily chosen maximum value. Even though  $\beta$  is never infinity, experimentally we observed that after certain value of  $\beta$  increasing it further does not change the solution, perhaps because of the finite machine precision.

**Convergence of MD in the Nonconvex Setting.** Even though MD is originally developed for convex optimization (similar to gradient descent), in this paper we directly apply MD to NN quantization where the loss is highly nonconvex and gradient estimates are stochastic. Nevertheless, MD converges in all our experiments while obtaining superior performance compared to the baselines. Theoretical analysis of MD for nonconvex, stochastic setting is an active research area [39,40] and MD has been recently shown to converge in the nonconvex stochastic setting under certain conditions [36]. We believe, similar to Theorem 2, the convergence analysis in [36] can be extended to MD with adaptive mirror maps, which goes beyond the scope of this work.

### 3.2 Numerically Stable form of Mirror Descent

We showed a few examples of valid projections, their corresponding mirror maps, and the final MD updates. Even though, in theory, these updates can be used directly, they are sometimes numerically unstable due to the operations involving multiple logarithms, exponentials and divisions [17]. To this end, we provide a numerically stable way of performing MD by storing a set of auxiliary parameters during training.

A careful look at the Fig. 1 suggests that the MD update with the mirror map derived from Theorem 1 can be performed by storing auxiliary variables

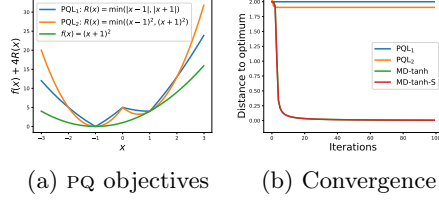


Fig. 3: *Convergence of PQ and MD when  $f(x) = (x+1)^2$  where  $\mathcal{X} = [-1, 1]$ , initialization  $\tilde{x}^0 = 1.2$  and  $\beta = 4$  (fixed) for all algorithms. While MD versions behave identically and converge to the optimal solution of  $-1$ , PQ versions get stuck at suboptimal (PQL<sub>2</sub> is nondiscrete) solutions.*

$\tilde{\mathbf{x}} = P^{-1}(\mathbf{x})$ . In fact, once the auxiliary variable  $\tilde{\mathbf{x}}^k$  is updated using gradient  $\mathbf{g}^k$ , it is directly mapped back to the constraint set  $\mathcal{X}$  via the projection. This is mainly because of the fact that the domain of the mirror maps derived based on the Theorem 1 are exactly the same as the constraint set. Formally, with this additional set of variables, one can write the MD update (4) corresponding to the projection  $P$  as:

$$\begin{aligned} \tilde{\mathbf{x}}^{k+1} &= \tilde{\mathbf{x}}^k - \eta \mathbf{g}^k, & \text{update in the dual space} \\ \mathbf{x}^{k+1} &= P(\tilde{\mathbf{x}}^{k+1}) \in \mathcal{X}, & \text{projection to the primal space} \end{aligned} \quad (23)$$

where  $\eta > 0$  and  $\mathbf{g}^k \in \partial f(\mathbf{x}^k)$ . Experimentally we observed these updates to show stable behaviour and performed remarkably well for both the tanh and softmax. We provide the pseudocode of this stable version of MD in Algorithm 2 for the tanh (MD-tanh-S). Extending it to other valid projections is trivial.

Note, above updates can be seen as optimizing the function  $f(P(\tilde{\mathbf{x}}))$  using gradient descent where the gradient through the projection (*i.e.*, Jacobian)  $J_P = \partial P(\tilde{\mathbf{x}})/\partial \tilde{\mathbf{x}}$  is replaced with the identity matrix. This is exactly the same as the Straight Through Estimator (STE) for NN quantization (following the nomenclature of [4,34]). Despite being a crude approximation, STE has shown to be highly effective for NN quantization with various network architectures and datasets [34,38]. However, a solid understanding of the effectiveness of STE is lacking in the literature except for its convergence analysis in certain cases [23,33]. In this work, by showing STE based gradient descent as an implementation method of MD under certain conditions on the projection, we provide a justification on the effectiveness of STE. Besides, as shown in Example 3, in cases where the MD formulation is nontrivial, the STE based implementation can be used.

### 3.3 Comparison against ProxQuant

The connection between the dual averaging version of MD and STE was recently hinted in ProxQuant (PQ) [4]. However, no analysis of whether an analogous mirror map exists to the given projection is provided and their final algorithm is not based on MD. In particular, following our notation, the final update equation of PQ can be written as:

$$\begin{aligned} \tilde{\mathbf{x}}^{k+1} &= \mathbf{x}^k - \eta \mathbf{g}^k, & \text{assumes } \mathbf{x}^k \text{ and } \mathbf{g}^k \text{ are in the same space} \\ \mathbf{x}^{k+1} &= \text{prox}(\tilde{\mathbf{x}}^{k+1}), & \text{prox} : \mathbb{R}^r \rightarrow \mathbb{R}^r \text{ is the proximal mapping defined in [4]} \end{aligned} \quad (24)$$

where  $\eta > 0$ , and  $\mathbf{g}^k \in \partial f(\mathbf{x}^k)$ . Note that, as opposed to MD (refer to Eq. (23)), PQ assumes the point  $\mathbf{x}^k$  and gradient  $\mathbf{g}^k$  are in the same space for the  $\tilde{\mathbf{x}}^{k+1}$  update to

be valid. This would only be true for the Euclidean space. However, as discussed in Sec. 2.1, MD allows gradient descent to be performed on a more general non-Euclidean space by first mapping the primal point  $\mathbf{x}^k$  to a point  $\tilde{\mathbf{x}}^k$  in the dual space via the mirror map. This is the core of MD, which allows faster convergence rates in certain cases, and enabled theoretical and practical research on MD for the past three decades.

Despite this fundamental difference, here we show that PQ can get stuck at a suboptimal (even nondiscrete) solution even in a simple convex setting. To this end, similarly to the convergence proof of PQ, we consider the case when the annealing hyperparameter  $\beta > 0$  (denoted by  $\lambda$  in [4]) is fixed. In this case, PQ optimizes the following objective:

$$\min_{\mathbf{x} \in \mathbb{R}^r} f(\mathbf{x}) + \beta R(\mathbf{x}) , \quad (25)$$

where the regularizer  $R$  is a “W” shaped nonconvex function. To this end, even when the loss function  $f$  is convex, the above composite PQ objective would be nonconvex and has multiple local minima for a range of values of  $\beta$ . Therefore the PQ algorithm is prone to converge to any of these local minima (which could be nondiscrete). Whereas, our MD algorithms are guaranteed to converge to the global optimum in the convex setting regardless of the value of  $\beta$ . This phenomenon is illustrated with a simple example in Fig. 3.

## 4 Related Work

In this work we consider parameter quantization, which is usually formulated as a constrained problem and optimized using a modified projected gradient descent algorithm, where the methods [3,4,9,10,11,32,34] mainly differ in the constraint set, the projection used, and how backpropagation through the projection is performed. Among them, STE based gradient descent is the most popular method as it enables backpropagation through nondifferentiable projections and it has shown to be highly effective in practice [11]. In fact, the success of this approach lead to various extensions by including additional layerwise scalars [29], relaxing the solution space [34], and even to quantizing activations [19], and/or gradients [38]. Moreover, there are methods focusing on loss aware quantization [16], quantization for specialized hardware [12], and quantization based on the variational approach [2,24,25]. We have only provided a brief summary of relevant methods and for a comprehensive survey we refer the reader to [14].

## 5 Experiments

Due to the popularity of binary neural networks [11,29], we mainly consider binary quantization and set the quantization levels as  $\mathcal{Q} = \{-1, 1\}$ . We would like to point out that we quantize all learnable parameters, meaning our approach results in *32 times less memory* compared to the floating-point counterparts.

We evaluate two MD variants corresponding to tanh and softmax projections, on CIFAR-10, CIFAR-100, TinyImageNet<sup>5</sup> and ImageNet datasets with VGG-16, ResNet-18 and MobileNetV2 architectures. We also evaluate the numerically stable versions of our MD variants (denoted with suffix “-s”) performed by storing auxiliary parameters during training as explained in Eq. (23). The results are compared against parameter quantization methods, namely BinaryConnect (BC) [11], PQ [4] and Proximal Mean-Field (PMF) [3]. In addition, for completeness, we also compare against a standard PGD variant corresponding to the tanh projection (denoted as GD-tanh), *i.e.*, minimizing  $f(\tanh(\tilde{\mathbf{x}}))$  using gradient descent. The only difference of this to our MD-tanh-s is that, in Eq. (23), the Jacobian of tanh is directly used in the updates. Note that, numerous techniques have emerged with BC as the workhorse algorithm by relaxing constraints such as the layer-wise scalars [29], and similar extensions are straightforward even in our case. Briefly, our results indicate that the binary networks obtained by the MD variants outperform comparable baselines yielding state-of-the-art performance.

For all the experiments, standard multi-class cross-entropy loss is used. We crossvalidate the hyperparameters such as learning rate, learning rate scale, rate of increase of annealing hyperparameter  $\beta$ , and their respective schedules for all tested methods including the baselines. This extensive crossvalidation improved the accuracies of previous methods by a large margin, *e.g.*, *up to 3% improvement for PMF*. We provide the hyperparameter tuning search space and the final hyperparameters in Appendix B. Our algorithm is implemented in PyTorch [28] and the experiments are performed on NVIDIA Tesla-P100 GPUs. Our code will be released upon publication.

## 5.1 Results

The classification accuracies of binary networks obtained by both variants of our algorithm, namely, MD-tanh and MD-softmax, their numerically stable versions (suffix “-s”) and the baselines BC, PQ, PMF, GD-tanh and the floating point Reference Network (REF) are reported in Table 1. Both the numerically stable MD variants consistently produce better or on par results compared to other binarization methods while narrowing the performance gap between binary networks and floating point counterparts to a large extent, on multiple datasets.

Our stable MD-variant perform slightly better than MD-softmax, whereas for tanh, MD updates either perform on par or sometimes even better than numerically stable version of MD-tanh. We believe, the main reason for this empirical variation in results for our MD-variants is due to numerical instability caused by the floating-point arithmetic of logarithm and exponential functions in Eq. (15) and Eq. (20). Furthermore, even though our two MD-variants, namely MD-softmax and MD-tanh optimize in different spaces, their performance is similar in most cases. This may be explained by the fact that both algorithms belong to the same family where a “soft” projection to the constraint set (in fact the constraints sets are equivalent in this case, refer Sec. 2.2) is used and an annealing hyperparameter is used to gradually enforce a quantized solution.

<sup>5</sup> <https://tiny-imagenet.herokuapp.com/>

Algorithm	Space	CIFAR-10		CIFAR-100		TinyImageNet
		VGG-16	ResNet-18	VGG-16	ResNet-18	ResNet-18
REF (float)	<b>w</b>	93.33	94.84	71.50	76.31	58.35
BC	<b>w</b>	89.04	91.64	59.13	72.14	49.65
PQ	<b>w</b>	85.41	90.76	39.61	65.13	44.32
PQ*	<b>w</b>	90.11	92.32	55.10	68.35	49.97
PMF	<b>u</b>	90.51	92.73	61.52	71.85	51.00
PMF*	<b>u</b>	91.40	93.24	<b>64.71</b>	71.56	51.52
GD-tanh	<b>w</b>	91.47	93.27	60.67	71.46	51.43
Ours	MD-softmax	<b>u</b>	90.47	91.28	56.25	68.49
	MD-tanh	<b>w</b>	<b>91.64</b>	92.97	61.31	72.13
	MD-softmax-s	<b>u</b>	91.30	<b>93.28</b>	63.97	<b>72.18</b>
	MD-tanh-s	<b>w</b>	91.53	93.18	61.69	<b>72.18</b>

Table 1: *Classification accuracies on the test set with full binary quantization. PQ\* denotes performance with biases, fully-connected layers, and shortcut layers in float (original PQ setup) whereas PQ represents full quantization. PMF\* denotes the performance of PMF after crossvalidation similar to our MD-variants and the original results from the paper are denoted as PMF. Note our MD variants obtained accuracies virtually the same as the best performing method and it outperformed the best method by a large margin in much harder TinyImageNet dataset.*

Note, PQ [4] does not quantize the fully-connected layers, biases and shortcut layers. For fair comparison, we crossvalidate PQ with all layers binarized and original PQ settings, and report the results denoted as PQ and PQ\* respectively in Table 1. Our MD-variants outperform PQ consistently on multiple datasets in equivalent experimental settings. This clearly shows that entropic or tanh-based regularization with our annealing scheme is superior to a simple “W” shaped regularizer and emphasizes that MD is a suitable framework for quantization.

Furthermore, the superior performance of MD-tanh against GD-tanh and on par or better performance of MD-softmax against PMF for binary quantization empirically validates that MD is useful even in a nonconvex stochastic setting. This hypothesis along with our numerically stable form of MD can be particularly useful to explore other projections which are useful for quantization and/or network compression in general.

The training curves for our MD variants for CIFAR-10 and CIFAR-100 datasets with ResNet-18 are shown in Fig. 4. The original MD variants show unstable behaviour during training which is attributed to the fact that it involves logarithms and exponentials in the update rules. In addition, we believe, the additional annealing hyperparameter also contributes to this instability. Regardless, by storing auxiliary variables, the MD updates are demonstrated to be quite stable. This clear distinction between MD variants emphasizes the significance of practical considerations while implementing MD especially in NN optimization.

To demonstrate the superiority of MD, we tested on a more resource efficient MobileNetV2 [30] and the results are summarized in Table 2. In short, our MD

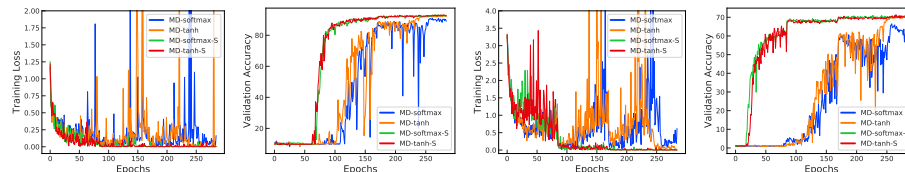


Fig. 4: *Training curves for binarization for CIFAR-10 (first two) and CIFAR-100 (last two) with ResNet-18. Compared to original MD variants, stable MD variants are less noisy and after the initial exploration phase (up to 60 in CIFAR-10 and 25 epochs CIFAR-100), the validation accuracies rise sharply and show gradual improvement afterwards.*

Algorithm	CIFAR-10	CIFAR-100
REF (float)	93.67	73.97
BC	86.84	65.04
$\frac{1}{2}$ MD-softmax-S	89.71	66.40
Ours MD-tanh-S	<b>89.99</b>	<b>66.63</b>

Table 2: *Classification accuracies on the test for MobileNetV2 where all the learnable parameters are binarized. Our stable MD variants significantly outperformed BC while MD-tanh-S is slightly better than MD-softmax-S in these cases.*

variants are able to fully-quantize MobileNetV2 with minimal loss in accuracy on CIFAR datasets. For more experiments such as training curves comparison to other methods and ternary quantization results please refer to the Appendix B.

**ImageNet.** We evaluated our MD-tanh-S against state-of-the-art methods on ImageNet with ResNet-18 and the results are reported in Table 3. Similar to all the compared algorithms, if we do not quantize the first convolution layer, last fully-connected layer, biases and batchnorm parameters, we set a *new state-of-the-art for binarization* with achieving merely  $< 4\%$  reduction compared to the floating-point network. Note that, the standard practice to quantize on ImageNet is to use floating-point scalars in each layer [29,32], however, our method outperforms all the methods without requiring layerwise scalars. In addition, MD-tanh-S yields the best results even when trained from scratch with  $< 1\%$  reduction compared to finetuning from a pretrained network. Furthermore, MD enables training of fully-binarized networks with no additional scalars (except the batchnorm parameters) from scratch, which is considered to be difficult for ImageNet [29]. More ablation study experiments can be found in Appendix B.

## 6 Discussion

In this work, we have introduced an MD framework for NN quantization by deriving mirror maps corresponding to various projections useful for quantization and provided a convergence analysis in the convex setting for time-varying mirror maps. In addition, we have discussed a numerically stable implementation of MD by storing an additional set of auxiliary variables and showed that this update is strikingly analogous to the popular STE based gradient method. The superior performance of our MD formulation even with simple projections such as tanh and softmax is encouraging and we believe, MD would be a suitable framework

Accuracy	REF	BWN*	BR	ELQ	ADMM	QN	Ours		
	(float)	[29]	[34]	[37]	[22]	[32]	MD-tanh-s* <sup>+</sup>	MD-tanh-s*	MD-tanh-s
top-1	70.61	60.80	63.20	64.72	64.80	66.50	56.67	65.92	<b>66.78</b>
top-5	89.40	83.00	85.10	86.04	86.20	<b>87.03</b>	79.66	86.29	87.01

Table 3: *ImageNet classification accuracies for binary quantization on ResNet-18. Here, \* indicates training from scratch and <sup>+</sup> indicates full-binarization except the batchnorm parameters. Note MD-tanh-s outperforms all other methods setting new state-of-the-art on ImageNet binarization. It might seem that the improvement over QN is marginal, however, QN requires layerwise scalars and pretraining [32], whereas MD-tanh-s does not require layerwise scaling and obtains near state-of-the-art results even without pretraining.*

for not just NN quantization but for network compression in general. In future, we intend to focus more on the theoretical aspects of MD in conjunction with stochastic momentum based optimizers such as Adam.

## 7 Acknowledgements

This work was supported by the ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1, EPSRC/MURI grant EP/N019474/1 and the Australian Research Council Centre of Excellence for Robotic Vision (project number CE140100016). We would like to thank Pawan Kumar for useful discussions on mirror descent and we acknowledge the Royal Academy of Engineering, FiveAI, Data61, CSIRO, and National Computing Infrastructure, Australia.

## Appendices

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

### A MD Proofs and Derivations

#### A.1 Deriving Mirror Maps from Projections

**Theorem 3.** *Let  $\mathcal{X}$  be a compact convex set and  $P : \mathbb{R} \rightarrow \mathcal{C}$  be an invertible function where  $\mathcal{C} \subset \mathbb{R}$  is a convex open set such that  $\mathcal{X} = \bar{\mathcal{C}}$  ( $\bar{\mathcal{C}}$  denotes the closure of  $\mathcal{C}$ ). Now, if*

1.  *$P$  is strictly monotonically increasing.*
2.  *$\lim_{x \rightarrow \partial \mathcal{C}} \|P^{-1}(x)\| = \infty$  ( $\partial \mathcal{C}$  denotes the boundary of  $\mathcal{C}$ ).*

*Then,  $\Phi(x) = \int_{x_0}^x P^{-1}(y)dy$  is a valid mirror map.*

*Proof.* From the fundamental theorem of calculus, the gradient of  $\Phi(x)$  satisfies,  $\nabla \Phi(x) = P^{-1}(x)$ . Since  $P$  is strictly monotonically increasing and invertible,  $P^{-1}$  is strictly monotonically increasing. Therefore,  $\Phi(x)$  is strictly convex and differentiable. Now, from the definition of projection and since it is invertible (*i.e.*,  $P^{-1}$  is *one-to-one* and *onto*),  $\nabla \Phi(\mathcal{C}) = P^{-1}(\mathcal{C}) = \mathbb{R}$ . Therefore, together with condition (2), we can conclude that  $\Phi(x) = \int_{x_0}^x P^{-1}(y)dy$  is a valid mirror map (refer Definition 1 in the main paper). For the multi-dimensional case, we need an additional condition that the vector field  $P^{-1}(\mathbf{x})$  is conservative. Then by the gradient theorem [1], there exists a mirror map  $\Phi(\mathbf{x}) = \int_{\mathbf{x}_0}^{\mathbf{x}} P^{-1}(\mathbf{y})d\mathbf{y}$  for some arbitrary base point  $\mathbf{x}_0$ .

#### A.2 MD Update Derivation for the tanh Projection

We now derive the MD update corresponding to the tanh projection below. From Theorem 3, the mirror map for the tanh projection can be written as:

$$\Phi(w) = \int P^{-1}(w)dw = \frac{1}{2\beta} [(1+w) \log(1+w) + (1-w) \log(1-w)] . \quad (26)$$

Correspondingly, the Bregman divergence can be written as:

$$\begin{aligned} D_{\Phi}(w, v) &= \Phi(w) - \Phi(v) - \Phi'(v)(w - v) , \quad \text{where } \Phi'(v) = \frac{1}{2\beta} \log \frac{1+v}{1-v} , \quad (27) \\ &= \frac{1}{2\beta} \left[ w \log \frac{(1+w)(1-v)}{(1-w)(1+v)} + \log(1-w)(1+w) - \log(1-v)(1-v) \right] . \end{aligned}$$

Now, consider the proximal form of MD update

$$w^{k+1} = \underset{\mathbf{x} \in (-1,1)}{\operatorname{argmin}} \langle \eta g^k, w \rangle + D_{\Phi}(w, w^k) . \quad (28)$$



The idea is to find  $w$  such that the KKT conditions are satisfied. To this end, let us first write the Lagrangian of Eq. (28) by introducing dual variables  $y$  and  $z$  corresponding to the constraints  $w > -1$  and  $w < 1$ , respectively:

$$F(w, x, y) = \eta g^k w + y(-w - 1) + z(w - 1) \quad (29)$$

$$+ \frac{1}{2\beta} \left[ w \log \frac{(1+w)(1-w^k)}{(1-w)(1+w^k)} + \log(1-w)(1+w) - \log(1-w^k)(1-w^k) \right] .$$

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

$$\frac{\partial F}{\partial w} = \eta g^k + \frac{1}{2\beta} \log \frac{(1+w)(1-w^k)}{(1-w)(1+w^k)} - y + z = 0 . \quad (30)$$

From complementary slackness conditions,

$$\begin{aligned} y(-w - 1) &= 0 , & \text{since } w > -1 \Rightarrow y &= 0 , \\ z(w - 1) &= 0 , & \text{since } w < 1 \Rightarrow z &= 0 . \end{aligned} \quad (31)$$

Therefore, Eq. (30) now simplifies to:

$$\begin{aligned} \frac{\partial F}{\partial w} &= \eta g^k + \frac{1}{2\beta} \log \frac{(1+w)(1-w^k)}{(1-w)(1+w^k)} = 0 , & (32) \\ \log \frac{(1+w)(1-w^k)}{(1-w)(1+w^k)} &= -2\beta \eta g^k , \\ \frac{1+w}{1-w} &= \frac{1+w^k}{1-w^k} \exp(-2\beta \eta g^k) , \\ w &= \frac{\frac{1+w^k}{1-w^k} \exp(-2\beta \eta g^k) - 1}{\frac{1+w^k}{1-w^k} \exp(-2\beta \eta g^k) + 1} . \end{aligned}$$

The pseudocodes of original (MD-tanh) and numerically stable versions (MD-tanh-s) for tanh are presented in Algorithms 1 and 2 respectively.

### A.3 Convergence Proof for MD with Adaptive Mirror Maps

**Theorem 4.** *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  $\Phi : \mathcal{C} \rightarrow \mathbb{R}$  be a mirror map  $\rho$ -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  $f : \mathcal{X} \rightarrow \mathbb{R}$  be a convex function and  $L$ -Lipschitz with respect to  $\|\cdot\|$ . Then MD with mirror map  $\Phi_\beta(\mathbf{x}) = \Phi(\mathbf{x})/\beta$  with  $1 \leq \beta \leq B$  and  $\eta = \frac{R}{L} \sqrt{\frac{2\rho}{Bt}}$  satisfies*

$$f\left(\frac{1}{t} \sum_{k=0}^{t-1} \mathbf{x}^k\right) - f(\mathbf{x}^*) \leq RL \sqrt{\frac{2B}{\rho t}} , \quad (33)$$

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

**Algorithm 1** MD-tanh**Require:**  $K, b, \{\eta^k\}, \rho > 1, \mathcal{D}, L$ **Ensure:**  $\mathbf{w}^* \in \mathcal{Q}^m$ 


---

```

1:  $\mathbf{w}^0 \in \mathbb{R}^m, \quad \beta \leftarrow 1$  ▷ Initialization
2:  $\mathbf{w}^0 \leftarrow \tanh(\beta \mathbf{w}^0)$  ▷ Projection
3: for  $k \leftarrow 0, \dots, K$  do
4:    $\mathcal{D}^b = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^b \sim \mathcal{D}$  ▷ Sample a mini-batch
5:    $\mathbf{g}^k \leftarrow \nabla_{\mathbf{w}} L(\mathbf{w}; \mathcal{D}^b)|_{\mathbf{w}=\mathbf{w}^k}$  ▷ Gradient w.r.t.  $\mathbf{w}$  at  $\mathbf{w}^k$  (Adam based gradients)
6:   for  $j \leftarrow 1, \dots, m$  do
7:      $w_j^{k+1} \leftarrow \frac{\frac{1+w_j^k}{1-w_j^k} \exp(-2\beta\eta^k g_j^k) - 1}{\frac{1+w_j^k}{1-w_j^k} \exp(-2\beta\eta^k g_j^k) + 1}$  ▷ MD update
8:   end for
9:    $\beta \leftarrow \rho\beta$  ▷ Increase  $\beta$ 
10: end for
11:  $\mathbf{w}^* \leftarrow \text{sign}(\tilde{\mathbf{w}}^K)$  ▷ Quantization

```

---

**Algorithm 2** MD-tanh-S**Require:**  $K, b, \{\eta^k\}, \rho > 1, \mathcal{D}, L$ **Ensure:**  $\mathbf{w}^* \in \mathcal{Q}^m$ 


---

```

1:  $\tilde{\mathbf{w}}^0 \in \mathbb{R}^m, \quad \beta \leftarrow 1$  ▷ Initialization
2: for  $k \leftarrow 0, \dots, K$  do
3:    $\mathbf{w}^k \leftarrow \tanh(\beta \tilde{\mathbf{w}}^k)$  ▷ Projection
4:    $\mathcal{D}^b = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^b \sim \mathcal{D}$  ▷ Sample a mini-batch
5:    $\mathbf{g}^k \leftarrow \nabla_{\mathbf{w}} L(\mathbf{w}; \mathcal{D}^b)|_{\mathbf{w}=\mathbf{w}^k}$  ▷ Gradient w.r.t.  $\mathbf{w}$  at  $\mathbf{w}^k$  (Adam based gradients)
6:    $\tilde{\mathbf{w}}^{k+1} \leftarrow \tilde{\mathbf{w}}^k - \eta^k \mathbf{g}^k$  ▷ Gradient descent on  $\tilde{\mathbf{w}}$ 
7:    $\beta \leftarrow \rho\beta$  ▷ Increase  $\beta$ 
8: end for
9:  $\mathbf{w}^* \leftarrow \text{sign}(\tilde{\mathbf{w}}^K)$  ▷ Quantization

```

---

*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 [7] for step by step derivation. Only a sketch is provided here. For the standard MD the bound is:

$$f\left(\frac{1}{t} \sum_{k=0}^{t-1} \mathbf{x}^k\right) - f(\mathbf{x}^*) \leq RL \sqrt{\frac{2}{\rho t}}, \quad (34)$$

with  $\eta = \frac{R}{L} \sqrt{\frac{2\rho}{t}}$ . Here, since  $\beta \leq B$ , the adaptive mirror map  $\Phi_\beta(\mathbf{x}) = \Phi(\mathbf{x})/\beta$  is  $\rho/B$ -strongly convex for all  $\beta$ . Therefore, by simply replacing  $\rho$  with  $\rho/B$  the desired bound is obtained. Since  $\beta \geq 1$ ,  $D_{\Phi_\beta}(\mathbf{x}, \mathbf{z}) = D_\Phi(\mathbf{x}, \mathbf{z})/\beta \leq D_\Phi(\mathbf{x}, \mathbf{z})$ , for all  $\mathbf{x}, \mathbf{z} \in \mathcal{C}$  and it is used in this proof.

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

$$\begin{aligned}\nabla\Phi_\beta(\mathbf{y}^{k+1}) &= \nabla\Phi_\beta(\mathbf{x}^k) - \eta \mathbf{g}^k, \quad \text{where } \mathbf{g}^k \in \partial f(\mathbf{x}^k) \text{ and } \mathbf{y}^{k+1} \in \mathcal{C}, \\ \mathbf{g}^k &= (\nabla\Phi_\beta(\mathbf{x}^k) - \nabla\Phi_\beta(\mathbf{y}^{k+1}))/\eta, \quad \eta > 0.\end{aligned}\quad (35)$$

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

$$\begin{aligned}f(\mathbf{x}^k) - f(\mathbf{x}) &\leq \langle \mathbf{g}^k, \mathbf{x}^k - \mathbf{x} \rangle, \quad f \text{ is convex}, \\ &= \langle \nabla\Phi_\beta(\mathbf{x}^k) - \nabla\Phi_\beta(\mathbf{y}^{k+1}), \mathbf{x}^k - \mathbf{x} \rangle / \eta, \quad \text{Eq. (35)}, \\ &= (D_{\Phi_\beta}(\mathbf{x}, \mathbf{x}^k) + D_{\Phi_\beta}(\mathbf{x}^k, \mathbf{y}^{k+1}) - D_{\Phi_\beta}(\mathbf{x}, \mathbf{y}^{k+1})) / \eta, \quad \text{Bregman div.}, \\ &\leq (D_{\Phi_\beta}(\mathbf{x}, \mathbf{x}^k) + D_{\Phi_\beta}(\mathbf{x}^k, \mathbf{y}^{k+1}) - D_{\Phi_\beta}(\mathbf{x}, \mathbf{x}^{k+1}) - D_{\Phi_\beta}(\mathbf{x}^{k+1}, \mathbf{y}^{k+1})) / \eta.\end{aligned}\quad (36)$$

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

$$\begin{aligned}\sum_{k=0}^{t-1} D_{\Phi_\beta}(\mathbf{x}, \mathbf{x}^k) - D_{\Phi_\beta}(\mathbf{x}, \mathbf{x}^{k+1}) &= \sum_{k=0}^{t-1} (D_{\Phi}(\mathbf{x}, \mathbf{x}^k) - D_{\Phi}(\mathbf{x}, \mathbf{x}^{k+1})) / \beta, \\ &\leq \sum_{k=0}^{t-1} D_{\Phi}(\mathbf{x}, \mathbf{x}^k) - D_{\Phi}(\mathbf{x}, \mathbf{x}^{k+1}), \quad \beta \geq 1, \\ &= D_{\Phi}(\mathbf{x}, \mathbf{x}^0) - D_{\Phi}(\mathbf{x}, \mathbf{x}^t), \\ &\leq D_{\Phi}(\mathbf{x}, \mathbf{x}^0), \quad D_{\Phi}(\mathbf{x}, \mathbf{z}) \geq 0, \quad \forall \mathbf{x}, \mathbf{z} \in \mathcal{C}.\end{aligned}\quad (37)$$

Now we bound the remaining term:

$$\begin{aligned}&D_{\Phi_\beta}(\mathbf{x}^k, \mathbf{y}^{k+1}) - D_{\Phi_\beta}(\mathbf{x}^{k+1}, \mathbf{y}^{k+1}) \\ &= \Phi_\beta(\mathbf{x}^k) - \Phi_\beta(\mathbf{x}^{k+1}) - \langle \nabla\Phi_\beta(\mathbf{y}^{k+1}), \mathbf{x}^k - \mathbf{x}^{k+1} \rangle, \quad \text{Bregman divergence def.}, \\ &\leq \langle \nabla\Phi_\beta(\mathbf{x}^k) - \nabla\Phi_\beta(\mathbf{y}^{k+1}), \mathbf{x}^k - \mathbf{x}^{k+1} \rangle - \frac{\rho}{2\beta} \|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2, \quad \Phi \text{ is } \rho\text{-strongly convex}, \\ &= \langle \eta \mathbf{g}^k, \mathbf{x}^k - \mathbf{x}^{k+1} \rangle - \frac{\rho}{2\beta} \|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2, \quad \text{Eq. (35)}, \\ &\leq \eta L \|\mathbf{x}^k - \mathbf{x}^{k+1}\| - \frac{\rho}{2\beta} \|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2, \quad f \text{ is } L\text{-Lipschitz}, \\ &\leq \frac{(\eta L)^2 \beta}{2\rho}, \quad az - bz^2 \leq a^2/(4b), \quad \forall z \in \mathbb{R}, \\ &\leq \frac{(\eta L)^2 B}{2\rho}, \quad \beta \leq B.\end{aligned}\quad (38)$$

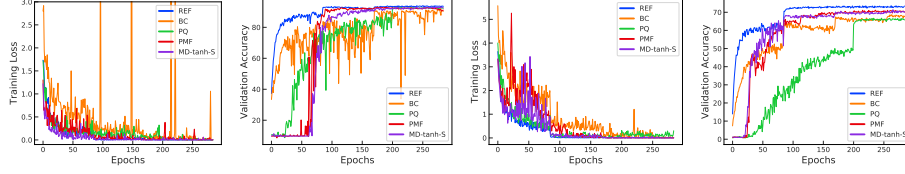


Fig. 5: Training curves for binarization for CIFAR-10 (first two columns) and CIFAR-100 (last two columns) with ResNet-18. Compared to BC, our MD-tanh-S and PMF are less noisy and after the initial exploration phase (up to 60 in CIFAR-10 and 25 epochs CIFAR-100), the validation accuracies rise sharply and closely resembles the floating point network afterwards. This steep increase is not observed in regularization methods such as PQ.

Putting Eqs. (37) and (38) in Eq. (36),

$$\begin{aligned}
 \frac{1}{t} \sum_{k=0}^{t-1} (f(\mathbf{x}^k) - f(\mathbf{x})) &\leq \frac{D_{\Phi}(\mathbf{x}, \mathbf{x}^0)}{\eta t} + \frac{\eta B L^2}{2\rho}, \\
 f\left(\frac{1}{t} \sum_{k=0}^{t-1} \mathbf{x}^k\right) - f(\mathbf{x}) &\leq \frac{R^2}{\eta t} + \frac{\eta B L^2}{2\rho}, \quad \text{Jensen inequality, defs. of } \mathbf{x}^0 \text{ and } R, \\
 &= R L \sqrt{\frac{2B}{\rho t}}, \quad \text{Substituting } \eta = \frac{R}{L} \sqrt{\frac{2\rho}{Bt}}.
 \end{aligned} \tag{39}$$

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

## 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. 5. Notice, after the initial exploration phase (due to low  $\beta$ ) 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.

### B.2 ImageNet Ablation Study

We provide an ablation study for various experimental settings for binary quantization on ImageNet dataset using ResNet-18 architecture in Table 4. We perform

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

Table 4: *Ablation study on ImageNet with ResNet-18 for 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.*

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

Table 5: *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	$32 \times 32$	10	45k / 5k	128	100k
CIFAR-100	$32 \times 32$	100	45k / 5k	128	100k
TinyImageNet	$64 \times 64$	200	100k / 10k	128	100k
ImageNet	$224 \times 224$	1000	1.2M / 50k	2048/256	90/55

Table 6: *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 from 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 7: *The hyperparameter search space for all the experiments. Chosen parameters are given in Tables 8, 9 and 10.*

### 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 5. 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<sup>rd</sup> quantization level. In contrast to us, the baseline method PQ [4] 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 [3]. To this end, the details of the datasets and their corresponding experiment setups are given in Table 6. For CIFAR-10/100 and TinyImageNet, VGG-16 [31], ResNet-18 [15] and MobileNetV2 [30] architectures adapted for CIFAR dataset are used. In particular, for CIFAR experiments, similar to [21], 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 [18]. In all the models, batch normalization [20] (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  $\beta$  and its schedule are crossvalidated from the range reported in Table 7 and the chosen parameters are given in the Table 8, Table 9 and Table 10. To generate the plots, we used the publicly available codes of BC<sup>6</sup>, PQ<sup>7</sup> and PMF<sup>8</sup>.

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  $\beta$  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.

**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 11. 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.

<sup>6</sup> <https://github.com/itayhubara/BinaryNet.pytorch>

<sup>7</sup> <https://github.com/allenbai01/ProxQuant>

<sup>8</sup> <https://github.com/tajanthan/pmf>

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 8: *Hyperparameter settings used for the binary quantization experiments. Here, the learning rate is multiplied by lr\_scale after every 30k iterations and annealing hyperparameter ( $\beta$ ) 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.*



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 9: *Hyperparameter settings used for the ternary quantization experiments. Here, the learning rate is multiplied by lr\_scale after every 30k iterations and annealing hyperparameter ( $\beta$ ) 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 10: *Hyperparameter settings used for the binary quantization experiments. Here, the learning rate is multiplied by lr\_scale after every 30k iterations and annealing hyperparameter ( $\beta$ ) 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 11: *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 [13], 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

1. Gradient theorem. [https://en.wikipedia.org/wiki/Gradient\\_theorem](https://en.wikipedia.org/wiki/Gradient_theorem), accessed: 31-07-2019
2. Achterhold, J., Kohler, J.M., Schmeink, A., Genewein, T.: Variational network quantization. ICLR (2018)
3. Ajanthan, T., Dokania, P.K., Hartley, R., Torr, P.H.: Proximal mean-field for neural network quantization. ICCV (2019)
4. Bai, Y., Wang, Y.X., Liberty, E.: Proxquant: Quantized neural networks via proximal operators. ICLR (2019)
5. Beck, A., Teboulle, M.: Mirror descent and nonlinear projected subgradient methods for convex optimization. Operations Research Letters (2003)
6. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge university press (2009)
7. Bubeck, S.: Convex optimization: Algorithms and complexity. Foundations and Trends® in Machine Learning (2015)
8. Bubeck, S., Cesa-Bianchi, N., Kakade, S.M.: Towards minimax policies for online linear optimization with bandit feedback. Conference on Learning Theory (2012)
9. Carreira-Perpinán, M.A., Idelbayev, Y.: Model compression as constrained optimization, with application to neural nets. part ii: Quantization. NeurIPS Workshop on Optimization for Machine Learning (2017)
10. Chen, S., Wang, W., Pan, S.J.: MetaQuant: learning to quantize by learning to penetrate non-differentiable quantization. NeurIPS (2019)
11. Courbariaux, M., Bengio, Y., David, J.P.: Binaryconnect: Training deep neural networks with binary weights during propagations. NeurIPS (2015)
12. Esser, S.K., Appuswamy, R., Merolla, P.A., Arthur, J.V., Modha, D.S.: Backpropagation for energy-efficient neuromorphic computing. NeurIPS (2015)
13. Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., He, K.: Accurate, large minibatch sgd: Training imagenet in 1 hour. arXiv preprint arXiv:1706.02677 (2017)
14. Guo, Y.: A survey on methods and theories of quantized neural networks. arXiv preprint arXiv:1808.04752 (2018)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CVPR (2016)
16. Hou, L., Yao, Q., Kwok, J.T.: Loss-aware binarization of deep networks. ICLR (2017)
17. Hsieh, Y.P., Kavis, A., Rolland, P., Cevher, V.: Mirrored langevin dynamics. NeurIPS (2018)
18. Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J.E., Weinberger, K.Q.: Snapshot ensembles: Train 1, get m for free. ICLR (2017)
19. Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., Bengio, Y.: Quantized neural networks: Training neural networks with low precision weights and activations. JMLR (2017)
20. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. ICML (2015)
21. Lee, N., Ajanthan, T., Torr, P.H.S.: SNIP: Single-shot network pruning based on connection sensitivity. ICLR (2019)
22. Leng, C., Dou, Z., Li, H., Zhu, S., Jin, R.: Extremely low bit neural network: Squeeze the last bit out with ADMM. AAAI (2018)

23. Li, H., De, S., Xu, Z., Studer, C., Samet, H., Goldstein, T.: Training quantized nets: A deeper understanding. *NeurIPS* (2017)
24. Louizos, C., Ullrich, K., Welling, M.: Bayesian compression for deep learning. *NeurIPS* (2017)
25. Louizos, C., Reisser, M., Blankevoort, T., Gavves, E., Welling, M.: Relaxed quantization for discretized neural networks. *ICLR* (2019)
26. McDonnell, M.D.: Training wide residual networks for deployment using a single bit for each weight. *ICLR* (2018)
27. Nemirovsky, A.S., Yudin, D.B.: Problem complexity and method efficiency in optimization. (1983)
28. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in PyTorch (2017)
29. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: Xnor-net: Imagenet classification using binary convolutional neural networks. *ECCV* (2016)
30. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: inverted residuals and linear bottlenecks. *CVPR* (2018)
31. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *ICLR* (2015)
32. Yang, J., Shen, X., Xing, J., Tian, X., Li, H., Deng, B., Huang, J., Hua, X.s.: Quantization networks. *CVPR* (2019)
33. Yin, P., Lyu, J., Zhang, S., Osher, S., Qi, Y., Xin, J.: Understanding straight-through estimator in training activation quantized neural nets. *ICLR* (2019)
34. Yin, P., Zhang, S., Lyu, J., Osher, S., Qi, Y., Xin, J.: Binaryrelax: A relaxation approach for training deep neural networks with quantized weights. *SIIMS* (2018)
35. Zhang, R., Lin, L., Zhang, R., Zuo, W., Zhang, L.: Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. *TIP* (2015)
36. Zhang, S., He, N.: On the convergence rate of stochastic mirror descent for nonsmooth nonconvex optimization. *CoRR* (2018)
37. Zhou, A., Yao, A., Wang, K., Chen, Y.: Explicit loss-error-aware quantization for low-bit deep neural networks. *CVPR* (2018)
38. Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., Zou, Y.: Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *CoRR* (2016)
39. Zhou, Z., Mertikopoulos, P., Bambos, N., Boyd, S., Glynn, P.: Mirror descent in non-convex stochastic programming. *CoRR* (2017)
40. Zhou, Z., Mertikopoulos, P., Bambos, N., Boyd, S., Glynn, P.W.: Stochastic mirror descent in variationally coherent optimization problems. *NeurIPS* (2017)