BinaryRelax: A Relaxation Approach For Training Deep Neural Networks With Quantized Weights*

Penghang Yin†, Shuai Zhang‡, Jiancheng Lyu‡, Stanley Osher †, Yingyong Qi ‡, and Jack Xin ‡

Abstract. We propose BinaryRelax, a simple two-phase algorithm, for training deep neural networks with quantized weights. The set constraint that characterizes the quantization of weights is not imposed until the late stage of training, and a sequence of *pseudo* quantized weights is maintained. Specifically, we relax the hard constraint into a continuous regularizer via Moreau envelope, which turns out to be the squared Euclidean distance to the set of quantized weights. The pseudo quantized weights are obtained by linearly interpolating between the float weights and their quantizations. A continuation strategy is adopted to push the weights towards the quantized state by gradually increasing the regularization parameter. In the second phase, exact quantization scheme with a small learning rate is invoked to guarantee fully quantized weights. We test BinaryRelax on the benchmark CIFAR-10 and CIFAR-100 color image datasets to demonstrate the superiority of the relaxed quantization approach and the improved accuracy over the state-of-the-art training methods. Finally, we prove the convergence of BinaryRelax under an approximate orthogonality condition.

Key words. BinaryRelax, deep neural networks, quantization, continuous relaxation.

AMS subject classifications. 90C10, 90C26, 90C90

1. Introduction.

Deep neural networks (DNNs) have achieved remarkable success in computer vision, speech recognition, and natural language processing systems [14, 16, 15, 24]. There is thus a growing interest in deploying DNNs on low-power embedded systems with limited memory storage and computing power, such as cell phones and other battery-powered devices. However, DNNs typically require hundreds of megabytes of memory storage for the trainable full-precision floating-point parameters or weights, and need billions of FLOPs to make a single inference. This makes the deployment of DNNs impractical on portable devices. Recent efforts have been devoted to the training of DNNs with coarsely quantized weights which are represented using low-precision (8 bits or less) fixed-point arithmetic [11, 5, 17, 32, 33, 29, 31, 21, 1]. Quantized neural networks enable substantial memory savings and computation/power efficiency, while achieving competitive performance with that of full-precision DNNs. Moreover, quantized weights can exploit hardware-friendly bitwise operations and lead to dramatic acceleration at inference time.

The simplest way to perform quantization would be directly rounding the weights of a pre-trained full-precision network. But without re-training, this naive approach often leads to poor accuracy at bit-width under 8. From the perspective of optimization, the training of

*The second and third authors contributed equally.

Funding: The work was partially supported by NSF grants DMS-1522383, IIS-1632935, ONR grant N00014-16-1-2157, DOE grant DE-SC00183838, and AFOSR grant FA 9550-15-0073.

†Department of Mathematics, University of California at Los Angeles, Los Angeles, CA 90095. (yph@ucla.edu, sjo@math.ucla.edu).

‡Department of Mathematics, University of California at Irvine, Irvine, CA 92697. (szhang3@uci.edu, jianchel@uci.edu, yqi@uci.edu, jxin@math.uci.edu)

quantized networks can be abstracted as a constrained optimization problem of minimizing some empirical risk subject to a set constraint that characterizes the quantization of weights:

$$\min_{x\in\mathbb{R}^n} \; f(x) := \frac{1}{N}\sum_{i=1}^{N} \ell_i(x) \quad \text{subject to} \quad x \in \mathcal{Q}. \tag{1}$$

The problem has specific structures. Given a training sample of input I_i and label u_i, the corresponding training loss takes the form

$$\ell_i(x) = \ell(\sigma_l(x_l * \cdots \sigma_1(x_1 * I_i)) - u_i),$$

where $x = [x_1, \ldots, x_l]^\top$ and $x_i \in \mathbb{R}^{n_i}$ contains the weights in the i-th linear (fully-connected or convolutional) layer, σ_i is some element-wise nonlinear function. "$*$" denotes either matrix-vector product or convolution operation; reshaping is necessary to avoid mismatch in dimensions. For layer-wise quantization, the set $\mathcal{Q}$ takes the form of $\mathcal{Q}_1 \times \cdots \times \mathcal{Q}_l$, where $x_i \in \mathcal{Q}_i := \mathbb{R}_+ \times \{\pm q_1, \pm q_2, \ldots, \pm q_m\}^{n_i}$. Here $\mathbb{R}_+$ denotes the set of nonnegative real numbers and $0 \le q_1 < q_2 < \cdots < q_m$ represent the m quantization levels and are pre-determined. The weight vector in the i-th layer enjoys the factorization $x_i = s_i \cdot Q_i$ for some $Q_i \in \{\pm q_1, \pm q_2, \ldots, \pm q_m = 1\}^{n_i}$ and some layer-wise scalar $s_i \ge 0$. Note that s_i does not have to be low-precision. s_i is shared by all weights across the i-th linear layer and will be stored separately from the quantized numbers Q_i for deployment efficiency. The storage for the scaling factors is *negligible* as there are so few of them. Weight quantization has two special cases as follows.

- 1-bit binarization: $m = 1$ and $\mathcal{Q}_i = \mathbb{R}_+ \times \{\pm 1\}^{n_i}$. The storage of Q_i's only needs 1 bit for representing the signs. Compared to the full-precision model, we have $32\times$ memory savings.
- 2-bit ternarization: $m = 2$ and $\mathcal{Q}_i = \mathbb{R}_+ \times \{0, \pm 1\}^{n_i}$. The storage needs 2 bits for representing the signs and the zero. Therefore, it gives $16\times$ model compression rate.

The acceleration through low-bit weights is achieved by leveraging the distributive law during forward propagation. For example, propagation through the first linear layer yields the computation of

$$x_1 * I = (s_1 Q_1) * I = s_1 (Q_1 * I).$$

When Q_1 is under 1-bit or 2-bit representation, the computation of $Q_1 * I$ can be extremely fast as there are additions/subtractions involved only.

On the computational side, with sampled mini-batch gradient ∇f_k at the k-th iteration, the classical projected stochastic gradient descent (PSGD)

$$\begin{cases} y^{k+1} = x^k - \gamma_k \nabla f_k(x^k) \\ x^{k+1} = \mathrm{proj}_{\mathcal{Q}}(y^{k+1}), \end{cases} \tag{2}$$

performs poorly however, and gets stagnated when updated with a small learning rate γ_k. It is the quantization/projection of weights that "rounds off" small gradient updates and causes the plateau [18]. Instead of using the standard gradient step in (2), a hybrid gradient update

$$y^{k+1} = y^k - \gamma_k \nabla f_k(x^k) \tag{3}$$

was proposed in [5] and showed significantly improved accuracy. This modification of PSGD is refered as BinaryConnect. Despite the succinctness and effectiveness of BinaryConnect, its convergence still lacks of understanding. The only analysis so far appears in [18], under convexity assumption on the objective function f. Researchers have also explored different schemes for quantizing float weights, whether deterministic or stochastic [5, 23, 32, 17, 29, 21, 31]. But to our knowledge, all these methods maintain a sequence of purely quantized weights, if not the optimal, during the training.

In this paper, we propose a novel relaxed quantization approach called BinaryRelax, to explore more freely the non-convex landscape of the objective function of the DNNs under the discrete quantization constraint. We relax the set constraint into a continuous regularizer, which leads to a relaxed quantization update. Besides, we set an increasing regularization parameter, driving x^k slowly to the quantized state. When the training error stops decaying at small γ_k, we switch to exact quantization to get genuinely quantized weights as desired. By exploiting the structure of quantization set $\mathcal{Q}$, we prove the convergence of BinaryRelax in the non-convex setting, which naturally covers that of BinaryConnect. To our knowledge, *this is the first convergence proof of BinaryConnect under non-convexity assumption.*

The rest of the paper is organized as follows. In section 2, we introduce the proposed BinaryRelax method. In section 3, we benchmark CIFAR-10 and CIFAR-100 datasets and compare BinaryRelax with state-of-the-art methods to demonstrate the benefits of performing relaxed quantization. In section 4, we establish the convergence results. The concluding remarks are given in section 5. All technical proofs are provided in the appendix.

Notations. $\|\cdot\|$ denotes the Euclidean norm; $\|\cdot\|_1$ denotes the ℓ_1 norm; $\|\cdot\|_0$ counts the number of nonzero components. $\mathbf{0} \in \mathbb{R}^n$ represents the vector of zeros. For any vector $x \in \mathbb{R}^n$ and closed set $\mathcal{Q} \subset \mathbb{R}^n$,

$$\mathrm{proj}_{\mathcal{Q}}(x) := \arg\min_{z\in\mathcal{Q}} \|x - z\|$$

is the projection of x onto $\mathcal{Q}$, and

$$\mathrm{dist}(x, \mathcal{Q}) := \min_{z\in\mathcal{Q}} \|x - z\|$$

is the Euclidean distance between x and $\mathcal{Q}$. When $\mathcal{Q}$ is a subspace in $\mathbb{R}^n$, $x \perp \mathcal{Q}$ means that x is orthogonal to $\mathcal{Q}$. $\mathrm{sign}(x)$ is the signum function acting pointwise on x, i.e.,

$$\mathrm{sign}(x)_i := \begin{cases} 1 & \text{if } x_i > 0, \\ -1 & \text{if } x_i < 0, \\ 0 & \text{if } x_i = 0. \end{cases}$$

2. BinaryRelax.

Without loss of generality, we assume the set of quantized weights

$$\mathcal{Q} = \mathbb{R}_+ \times \{\pm q_1, \ldots, \pm q_m\}^n \subset \mathbb{R}^n$$

throughout the paper, that is, we only consider the case for simplicity that a single scaling factor is shared by all weights in the network.

2.1. Quantization. In fact, for b-bit quantization, $\mathcal{Q} = \mathcal{L}_1 \cup \cdots \cup \mathcal{L}_p$ is the union of p one-dimensional subspaces $\mathcal{L}_i$ in $\mathbb{R}^n$, $i = 1, \ldots, p$, where $p = 2^{n-1}$ for $b = 1$ and $p = \frac{(2^b-1)^n-1}{2}$ for $b \geq 2$, and

$$\mathcal{L}_i = \{s \cdot Q_i : s \in \mathbb{R},\ Q_i \in \{\pm q_1, \ldots, \pm q_m\}^n \setminus \{\mathbf{0}\}\}$$

with $Q_i \neq \pm Q_j$ for $i \neq j$.

Given the float weight vector y^k in the k-th iteration, the quantized weights x^k obtained from y^k is basically the projection of y^k onto the set $\mathcal{Q}$. Therefore, the quantization of y^k gives rise to the optimization problem

$$x^k = \arg\min_{x \in \mathcal{Q}} \|x - y^k\|^2 = \mathrm{proj}_{\mathcal{Q}}(y^k). \tag{4}$$

The above projection/quantization problem can be reformulated as

$$(s_+^k, Q^k) = \arg\min_{s_+, Q} \|s_+ \cdot Q - y^k\|^2 \quad \text{subject to} \quad s_+ \geq 0,\ Q \in \{\pm q_1, \ldots, \pm q_m\}^n, \tag{5}$$

which is essentially a constrained K-means clustering problem. The centroids are parameterized by a single parameter s_+. The assigned centroids or quantization is given by $x^k = s_+^k \cdot Q^k$.

It has been shown that the closed form (exact) solution of (5) can be computed at $O(n)$ complexity for binarization [23] where $Q \in \{\pm 1\}^n$:

$$s_+^k = \frac{\|y^k\|_1}{n},\ Q_i^k = \begin{cases} 1 & \text{if } y_i^k \geq 0 \\ -1 & \text{otherwise.} \end{cases} \tag{6}$$

In the case of ternarization where $Q \in \{0, \pm 1\}^n$, an $O(n \log n)$ exact formula was found in [29]:

$$t^* = \arg\max_{1 \leq t \leq n} \frac{\|y_{[t]}^k\|_1^2}{t},\ s_+^k = \frac{\|y_{[t^*]}^k\|_1}{t^*},\ Q^k = \mathrm{sign}(y_{[t^*]}^k), \tag{7}$$

where $y_{[t]} \in \mathbb{R}^n$ keeps the t largest component in magnitude of y, while zeroing out the others. For quantization with wider bit-width ($b > 2$), accurately solving (5) becomes computationally intractable [29]. Empirical formulas have thus been proposed for an approximate quantized solution [17, 29, 31], and they are sufficient for practical use. For example, a heuristic thresholding scheme of $O(n)$ complexity for ternarization was proposed in [17] as

$$\delta = \frac{0.7\|y^k\|_1}{n},\ s_+^k = \frac{\sum_{i=1}^n |y_i^k| \cdot 1_{|y_i^k| \geq \delta}}{\sum_{i=1}^n 1_{|y_i^k| \geq \delta}},\ Q_i^k = \begin{cases} \mathrm{sign}(y_i^k) & \text{if } |y_i^k| \geq \delta \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

The focus of this paper is not on how to quantize a float weight vector. So we simply assume that the quantization $\mathrm{proj}_{\mathcal{Q}}(y^k)$ can be computed precisely, regardless the choice of $\mathcal{Q}$.

2.2. Moreau envelope and proximal mapping. In the seminal paper [20], Moreau introduced what is now called the Moreau envelope and the proximity operator (a.k.a. proximal mapping) that generalizes the projection. Let $g : \mathbb{R}^n \to (-\infty, \infty]$ be a lower semi-continuous extended-real-valued function. For any $t > 0$, the Moreau envelope function g_t is defined by

$$g_t(x) := \inf_{z \in \mathbb{R}^n} \; g(z) + \frac{1}{2t}\|z - x\|^2.$$

In general, g_t is everywhere finite and locally Lipschitz continuous. Moreover, g_t converges pointwise to g as $t \to 0^+$. Moreau envelope is closely related to the inviscid Hamilton-Jacobi equation [4]

$$u_t + \frac{1}{2} \mid \nabla_x u \mid^2 = 0, \quad u(x, 0) = g(x),$$

where $u(x, t) = g_t(x)$ is the unique viscosity solution of the above initial-value problem via the Hopf-Lax formula

$$u(x, t) = \inf_z \left\{ f(z) + tH^* \left(\frac{z - x}{t} \right) \right\}$$

with the Hamiltonian $H(t, x, v) = \frac{1}{2}\|v\|^2$ and its Fenchel conjugate $H^* = H$.

The proximal mapping of g is defined by

$$\text{prox}_g(x) := \arg\min_{z \in \mathbb{R}^n} \; g(z) + \frac{1}{2}\|z - x\|^2.$$

It is frequently used in optimization algorithms associated with non-smooth optimization problems such as total variation denoising [9].

2.3. Relaxed quantization. Let us begin with the alternative form of DNNs quantization problem (1):

$$\min_{x \in \mathbb{R}^n} \; f(x) + \chi_{\mathcal{Q}}(x), \tag{9}$$

where $\chi_{\mathcal{Q}}(x)$ is the characteristic function of $\mathcal{Q}$ defined by

$$\chi_{\mathcal{Q}}(x) = \begin{cases} 0 & \text{if } x \in \mathcal{Q} \\ \infty & \text{otherwise.} \end{cases}$$

When both the objective function $f(x)$ and the set $\mathcal{Q}$ are non-convex, the discontinuity of $\chi_{\mathcal{Q}}$ poses an extra challenge in minimization since a continuous gradient descent update can be made stagnant when projected discontinuously. Since $\mathcal{Q}$ is closed and the characteristic function of any closed set is lower semi-continuous, the Moreau envelope of $\chi_{\mathcal{Q}}$ is well defined for $t > 0$ and is given by

$$\inf_z \; \chi_{\mathcal{Q}}(z) + \frac{1}{2t}\|z - x\|^2 = \frac{1}{2t}\text{dist}(x, \mathcal{Q})^2.$$

The (squared) distance function $\text{dist}(x, \mathcal{Q})^2$ is continuously differentiable almost everywhere, except at points that have at least two nearest line subspaces, i.e., there exist two different

ways to quantize x. We use $\frac{1}{2t}\mathrm{dist}(x,\mathcal{Q})^2$ as the approximant of the discontinuous $\chi_{\mathcal{Q}}(z)$ and propose to minimize the relaxed training error

$$\min_{x\in\mathbb{R}^n} \; f(x)+\frac{\lambda}{2}\mathrm{dist}(x,\mathcal{Q})^2, \tag{10}$$

where $\lambda = t^{-1} > 0$ is the regularization parameter. When $\lambda\to\infty$, $\frac{\lambda}{2}\mathrm{dist}(x,\mathcal{Q})^2$ converges pointwise to $\chi_{\mathcal{Q}}(x)$, and the global minimum of (10) converges to that of (9).

Proposition 2.1. *Suppose $f(x)$ is continuous. Let $f_{\mathcal{Q}}^* = \min_{x\in\mathcal{Q}} f(x)$ be the global minimum of (9) and x_λ^* be the global minimizer of relaxed quantization problem (10). Then*

$$\mathrm{dist}(x_\lambda^*,\mathcal{Q})\to 0 \text{ and } f(x_\lambda^*)\to f_{\mathcal{Q}}^*, \text{ as } \lambda\to\infty.$$

2.4. Algorithm. Inspired by the hybrid gradient update proposed in [5], we write a two-line solver for the minimization problem (10):

$$\begin{cases} y^{k+1} = y^k - \gamma_k\nabla f_k(x^k) \\ x^{k+1} = \arg\min_{x\in\mathbb{R}^n} \frac{1}{2}\|x-y^{k+1}\|^2 + \frac{\lambda}{2}\mathrm{dist}(x,\mathcal{Q})^2. \end{cases}$$

The algorithm constructs two sequences: an auxiliary sequence of float weights $\{y^k\}$ and a sequence of *nearly* quantized weights $\{x^k\}$. The mismatch of discontinuous projection and continuous gradient descent is resolved by the relaxed quantization step:

$$x^{k+1} = \arg\min_{x\in\mathbb{R}^n} \frac{1}{2}\|x-y^{k+1}\|^2 + \frac{\lambda}{2}\mathrm{dist}(x,\mathcal{Q})^2, \tag{11}$$

which calls for computing the proximal mapping of the function $\frac{\lambda}{2}\mathrm{dist}(x,\mathcal{Q})^2$. This can be done via a simple formula.

Proposition 2.2. *Let*

$$\mathrm{proj}_{\mathcal{Q}}(y^{k+1}) = \arg\min_{x\in\mathcal{Q}} \|x-y^{k+1}\|^2$$

be the accurate quantization of y^{k+1}, then the solution to relaxed quantization subproblem (11) is given by

$$x^{k+1} = \frac{\lambda\,\mathrm{proj}_{\mathcal{Q}}(y^{k+1}) + y^{k+1}}{\lambda+1}. \tag{12}$$

Note that we still need the exact quantization $\mathrm{proj}_{\mathcal{Q}}(y^{k+1})$ to perform relaxed quantization. The update x^{k+1} is essentially a linear interpolation between y^{k+1} and its quantization $\mathrm{proj}_{\mathcal{Q}}(y^{k+1})$, and λ controls the weighted average. x^{k+1} is not precisely quantized because $x^{k+1}\notin\mathcal{Q}$, but x^{k+1} approaches $\mathcal{Q}$ as λ increases. Hereby we adopt a continuation strategy and let λ grow exponentially, which may not be the best but gives satisfactory performance in our experiments. Specifically, we inflate λ in every epoch by a factor $\rho\gtrapprox 1$. Intuitively, the relaxation with continuation will help skip over some bad local minima of (9) located in $\mathcal{Q}$, because they are not local minima of the relaxed formulation in general.

Proposition 2.3. *Suppose $f(x)$ is differentiable. Any point $x^* \in \mathcal{Q}$ is not a local minimizer of the relaxed quantization problem (10) unless $\nabla f(x^*) = \mathbf{0}$.*

In order to obtain purely quantized weights in the end, we turn off the relaxation mode and enforce exact quantization

$$\begin{cases} y^{k+1} = y^k - \gamma_k \nabla f_k(x^k) \\ x^{k+1} = \mathrm{proj}_{\mathcal{Q}}(y^{k+1}). \end{cases} \tag{13}$$

The BinaryRelax algorithm is summarized in Alg. 1 below. In fact, the Phase II update (13) is not new, and it has become the workhorse for weight quantization of networks [5, 23, 17, 32, 29]. In a recent study [18], it was referred as the BinaryConnect scheme.

Algorithm 1 BinaryRelax.

Input: number of epochs for training, batch size, schedule of learning rate $\{\gamma_k\}$, growth factor $\rho \gtrapprox 1$.

```
for i = 1, 2,..., nb-epoch do
  Randomly shuffle the data and partition into batches.
  for j = 1, 2, ..., nb-batch do
    y^{k+1} = y^k − γ_k ∇f_k(x^k)
    if i ≤ T then
      x^{k+1} = (λ_k proj_Q(y^{k+1}) + y^{k+1}) / (λ_k + 1)    // Phase I
      λ_{k+1} = ρ λ_k
    else
      x^{k+1} = proj_Q(y^{k+1})    // Phase II
    end if
    k = k + 1
  end for
end for
```

Remark 2.4. The idea of relaxing the discrete sparsity constraint $\|x\|_0 \le s$ into a continuous and possibly non-convex regularizer has been long known in the contexts of statistics and compressed sensing [27, 8, 2]. For example, compressed sensing solvers for minimizing the convex ℓ_1 norm [9] or non-convex sparse proxies, such as $\ell_{1/2}$ (with smoothing) [3] and ℓ_{1-2} [28], often outperform those directly tackling the nonzero counting metric ℓ_0. Interestingly, similar to $\mathcal{Q}$, the sparsity constraint set $\{x \in \mathbb{R}^n : \|x\|_0 \le s\}$ is also a finite union of low-dimensional subspaces in $\mathbb{R}^n$.

Remark 2.5. BinaryConnect resembles the linearized Bregman algorithm proposed by Osher et al. [30] for solving the basis pursuit problem

$$\begin{cases} v^{k+1} = v^k - A^\top (Au^k - b) \\ u^{k+1} = \delta \cdot \mathrm{shrink}(v^{k+1}, \mu) \end{cases}$$

where δ, $\mu > 0$ are parameters. In linearized Bregman, $A^\top(Au - b)$ is the gradient of sum

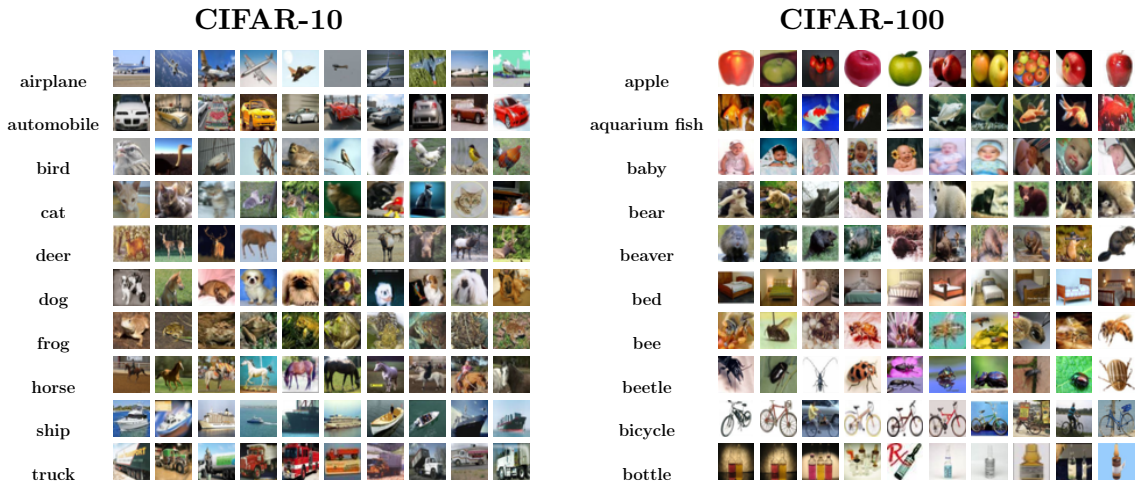

Figure 1. *Sample images from CIFAR datasets: 10 classes in CIFAR-10 (left); 10 out of 100 classes in CIFAR-100 (right).*

of squares $\frac{1}{2}\|Au-b\|^2$, and $\text{shrink}(v,\mu)$ is the proximal operator of ℓ_1 norm (a.k.a. soft-thresholding operator [7]):

$$\text{shrink}(v,\mu) := \arg\min_u \ \frac{1}{2\mu}\|u-v\|^2 + \|u\|_1.$$

3. Experimental Results. We tested BinaryRelax on benchmark CIFAR-10 and CIFAR-100 color image datasets [13], and compared with BinaryConnect on layer-wise binarization and ternarization. The CIFAR-10 dataset consists of 60000 32×32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. CIFAR-100 dataset is just like the CIFAR-10, except it has 100 classes containing 600 images each. There are 500 training images and 100 test images per class. Fig. 1 shows some sample images from CIFAR datasets. In the experiments, we used the testing images for validation. We coded up the BinaryRelax in PyTorch [22] platform. All experiments were carried out on two desktops with Nvidia graphics cards GTX 1080 Ti and Titan X.

The two baselines are the BinaryConnect framework (13) combined with the exact binarization formula (6) (BWN) [23] and the heuristic ternarization scheme (8) (TWN) [17], resp.. We used the same quantization formulas for BinaryRelax in the relaxed quantization update (12). Besides, we ran the same number of epochs, and the schedules of learning rate were also the same. We set the multi-step learning rates $\{0.1, 0.01, 0.001\}$ and the initial relaxation parameter $\lambda_0 = 1$ in all of our experiments. Phase II starts a few epochs after the learning rate decreases to 0.001 in the last stage of training. Then we find a proper growth factor ρ, such that $\lambda \in (100, 200)$ at the moment Phase I ends. In Phase II, BinaryRelax basically reduces to BWN or TWN. In addition, we used batch size = 128, ℓ_2 weight decay = 10^{-4}, batch normalization [12], and momentum = 0.95.

We tested the algorithms on the popular VGG [26] and ResNet[10] architectures, and the validation accuracies for CIFAR-10 and CIFAR-100 are summarized in Tab. 1 and Tab.

2, resp.. Note that ResNet-18 and ResNet-34 tested here were originally constructed for the more challenging ImageNet classification [6] and then adapted for CIFAR datasets. They have wider channels in the convolutional layers and are much larger than the other ResNets. For example, ResNet-18 has $\sim$ 11 million parameters, whereas ResNet-110 has only $\sim$ 1.7 million. This explains their higher accuracies. All quantized networks were initialized from their full-precision counterparts whose validation accuracies are listed in the second column. Fig. 2 shows the validation accuracies for CIFAR-100 tests with VGG-16 and ResNet-34 during the training process. With approximately the same training cost, our relaxed quantization approach consistently outperforms the hard quantization counterpart in validation accuracies. As seen from the tables and figure, the advantage of relaxed quantization is particularly clear when it comes to the large nets ResNet-18 and ResNet-34, where we have more complex landscapes with spurious local minima. In this case, our accuracies of binarized networks even surpass that of TWN. The relaxation indeed helps skip over bad local minima during the training.

CIFAR-10	Float	Binary		Ternary	
		BWN	Ours	TWN	Ours
VGG-11	91.93	88.70	89.28	90.48	91.01
VGG-16	93.59	91.60	91.98	92.75	93.20
ResNet-20	92.68	87.44	87.82	88.65	90.07
ResNet-32	93.40	89.49	90.65	90.94	92.04
ResNet-18	95.49	92.72	94.19	93.55	94.98
ResNet-34	95.70	93.25	94.66	94.05	95.07

Table 1
CIFAR-10 validation accuracies.

CIFAR-100	Float	Binary		Ternary	
		BWN	Ours	TWN	Ours
VGG-11	70.43	62.35	63.82	64.16	65.87
VGG-16	73.55	69.03	70.14	71.41	72.10
ResNet-56	70.86	66.73	67.65	68.26	69.83
ResNet-110	73.21	68.67	69.85	68.95	72.32
ResNet-18	76.32	72.31	74.04	73.15	75.24
ResNet-34	77.23	72.92	75.62	74.43	76.16

Table 2
CIFAR-100 validation accuracies.

4. Convergence Analysis. In this section, we analyze the convergence property of the proposed BinaryRelax. More precisely, we will focus on the iterations (13) in Phase II of

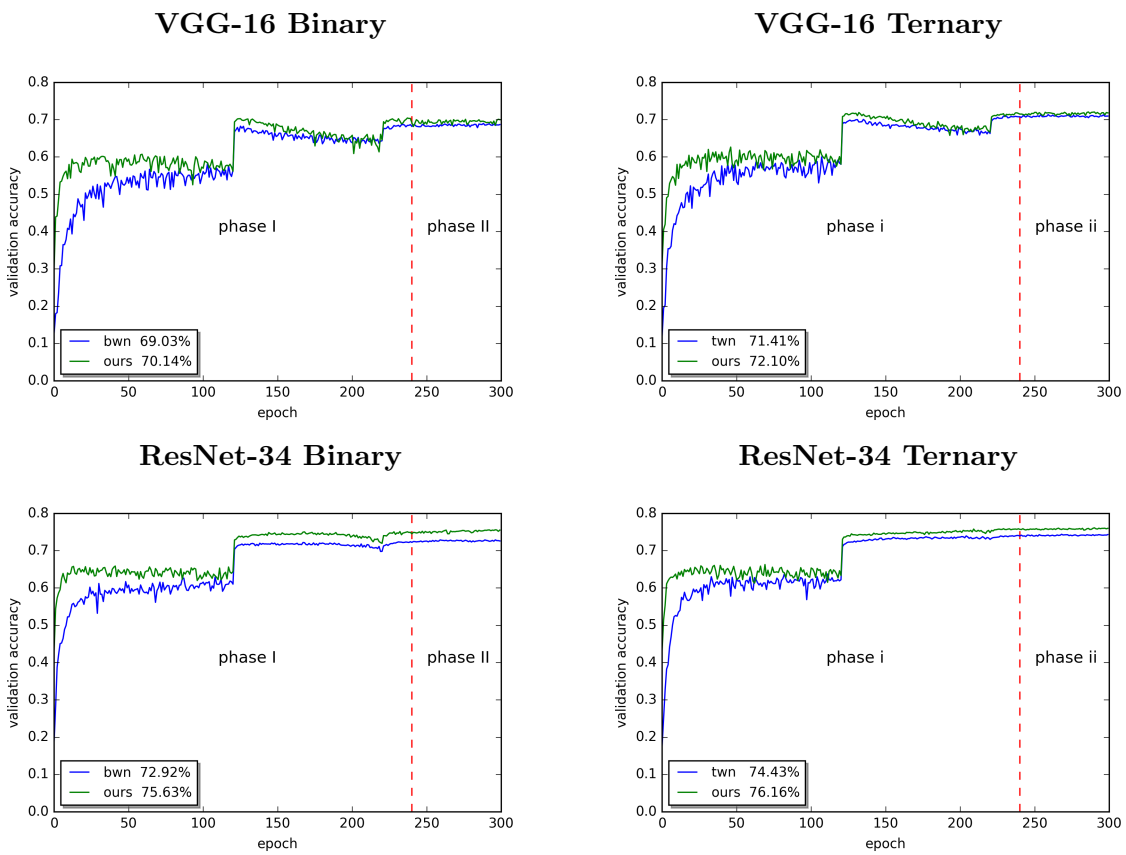

Figure 2. *Comparisons of validation accuracy curves for CIFAR-100 using VGG-16 and ResNet-34. The initial learning rate* $\gamma_0 = 0.1$ *and decays by a factor of* 0.1 *at epoch* 120 *and* 220. *The initial regularization parameter* $\lambda_0 = 1$ *and grows by a factor of* $\rho = 1.02$ *after each epoch until epoch* 240 *where Phase II starts.*

BinaryRelax (i.e., BinaryConnect):

$$\begin{cases} y^{k+1} = y^k - \gamma_k \nabla f_k(x^k) \\ x^{k+1} = \mathrm{proj}_{\mathcal{Q}}(y^{k+1}). \end{cases}$$

Although the convergence of BinaryConnect at a small learning rate is observed empirically (as seen from our experiments), the only convergence results, to our knowledge, were established in [18], in terms of the object value under convexity assumption made on f. However, the loss functions in deep learning are notoriously non-convex. Under more realistic assumptions, we shall show the sequence $\{x^k\}$ generated by Alg. 1 subsequentially converges in expectation to an approximate critical point. The convergence is established under a novel approximate orthogonality condition by exploiting the property of the set $\mathcal{Q}$ being the union of line subspaces $\mathcal{L}_i$. Therefore, our analysis cannot be readily extended to problems under general discrete constraint.

4.1. Preliminaries. We have the following basic assumptions.

Assumption 4.1. $f(x)$ is bounded from below. Without loss of generality, we assume the lower bound is 0.

Assumption 4.2. $f(x)$ is L-Lipschitz differentiable, i.e., for any $x, y \in \mathbb{R}^n$, we have

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|.$$

Assumption 4.3. $\mathbb{E}[\|\nabla f(x^k) - \nabla f_k(x^k)\|^2] \leq \sigma^2$ for all $k \in \mathbb{N}$, where the expectation is taken over the stochasticity of the algorithm.

Our proof relies on the following technical lemmata that exploit the structure of set $\mathcal{Q}$.

Lemma 4.4 (Approximate orthogonality). *Let $\{y^k\}, \{x^k\}$ be defined in Alg. 1. There exists $\alpha_k \geq 0$, such that*

$$\alpha_k\|x^{k+1} - x^k\|^2 + \|y^k - x^k\|^2 = \|y^k - x^{k+1}\|^2.$$

Proposition 4.5. *Let $\theta_{\min}$ be the smallest angle formed by any two line subspaces in $\mathcal{Q}$. If $\|x^{k+1} - x^k\| < \|x^k\| \sin\theta_{\min}$, then $\alpha_k = 1$ in Lemma 4.4. Moreover, α_k may have to be 0 only when $\|y^k - x^k\| = \|y^k - x^{k+1}\|$ and $\nabla f_k(x^k) \perp \mathcal{L}_i$ with $\mathcal{L}_i$ containing x^{k+1}.*

The above proposition implies that α_k is generally positive and approaches 1 when the relative change in consecutive iterates is getting small.

Lemma 4.6 (Alternative update). *Let $\{x^k\}$ be defined in Alg. 1. Suppose $x^{k+1} \in \mathcal{L}_i \subset \mathcal{Q}$ with $\mathcal{L}_i$ being some line subspace and define $\tilde{x}^k := \mathrm{proj}_{\mathcal{L}_i}(y^k)$, then*

$$x^{k+1} = \arg\min_{x \in \mathcal{L}_i} \|x - (\tilde{x}^k - \gamma_k \nabla f_k(x^k))\|^2.$$

Moreover, x^{k+1} is a local minimizer of the following problem

$$\min_{x \in \mathcal{Q}} \|x - (\tilde{x}^k - \gamma_k \nabla f_k(x^k))\|^2. \tag{14}$$

Lemma 4.7. *Let α_k and $\tilde{x}^k$ be defined in Lemma 4.4 and 4.6, resp., it holds that*

$$\|x^{k+1} - \tilde{x}^k\|^2 \leq \alpha_k\|x^{k+1} - x^k\|^2.$$

As always, the descent lemma is crucial for constructing the Lyapunov function.

Lemma 4.8 (Descent lemma). *For any x, y, it holds that*

$$f(x) \leq f(y) + \langle \nabla f(y), x - y\rangle + \frac{L}{2}\|x - y\|^2.$$

We recall the definition of subdifferential for proper and lower semicontinuous functions.

Definition 4.9 (Subdifferential [19, 25]). *Let $h : \mathbb{R}^n \to (-\infty, +\infty]$ be a proper and lower semicontinuous function. We define $\mathrm{dom}(h) := \{x \in \mathbb{R}^n : h(x) < +\infty\}$. For a given $x \in \mathrm{dom}(h)$, the Fréchet subdifferential of h at x, written as $\hat{\partial}h(x)$, is the set of all vectors $u \in \mathbb{R}^n$ which satisfy*

$$\lim_{y \neq x} \inf_{y \to x} \frac{h(y) - h(x) - \langle u, y - x\rangle}{\|y - x\|} \geq 0.$$

When $x \notin \mathrm{dom}(h)$, we set $\hat{\partial}h(x) = \emptyset$. The (limiting) subdifferential, or simply the subdifferential, of h at $x \in \mathbb{R}^n$, written as $\partial h(x)$, is defined through the following closure process

$$\partial h(x) := \{u \in \mathbb{R}^n : \exists\, x^k \to x,\ h(x^k) \to h(x) \text{ and } u^k \in \hat{\partial}h(x^k) \to u \text{ as } k \to \infty\}.$$

4.2. Main results. We are in the position to present the convergence results, which are established under an approximate orthogonality condition on α_k in Lemma 4.4.

Theorem 4.10. *Let $\{x^k\}$ be the sequence generated by Alg. 1. Suppose there exist $\underline{\alpha}, \bar{\alpha}, \gamma > 0$ such that $\underline{\alpha} \leq \alpha_k \leq \bar{\alpha}$ and $\gamma_{k+1} \leq \gamma_k \leq \gamma < \frac{\underline{\alpha}}{2L}$ for all $k \in \mathbb{N}$. Then*

$$\lim_{k\to\infty} \mathbb{E}[\|x^{k+1} - x^k\|^2] = 0,$$

if $\sum_{k=0}^{\infty} \gamma_k^2 < \infty$. If further $\sum_{k=0}^{\infty} \gamma_k = \infty$, we have

$$\liminf_{k\to\infty} \mathbb{E}[\mathrm{dist}(\mathbf{0}, \partial h(x^k))^2] \leq \frac{\sigma^2}{3}\left(\frac{4\bar{\alpha}}{\underline{\alpha}^2} + 1\right),$$

where $h = f + \chi_{\mathcal{Q}}$.

5. Concluding Remarks. From the view point of optimization, we proposed BinaryRelax, a novel relaxation approach based on Moreau envelope, for training quantized neural networks. Our algorithm iterates between a hybrid gradient step for updating the float weights and a weighted average of the computed float weights and their quantizations. We increase slowly the parameter that controls the average to drive the weights to the quantized state. In order to get the purely quantized weights, exact quantization replaces the weighted average in the second phase of training. Extensive experiments shows that with about the same training cost, BinaryRelax is consistently better than its BinaryConnect counterpart in terms of validation accuracy. It has clearer advantage on larger networks, which yield more complex landscape of the training loss with spurious local minima. In addition, BinaryRelax is provably convergent in expectation under an approximate orthogonality condition, which is another contribution of this paper.

REFERENCES

[1] Z. Cai, X. He, J. Sun, and N. Vasconcelos, *Deep learning with low precision by half-wave gaussian quantization*, Computer Vision and Pattern Recognition, IEEE Conference on, (2017).

[2] E. Candès, J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal rconstruction from highly incomplete frequency information*, IEEE Trans. Info. Theory, 52 (2006), pp. 489–509.

[3] R. Chartrand and W. Yin, *Iteratively reweighted algorithms for compressive sensing*, Acoustics, speech and signal processing, IEEE international conference on, (2008).

[4] P. Chaudhari, A. Oberman, S. Osher, S. Soatto, and G. Carlier, *Deep relaxation: partial differential equations for optimizing deep neural networks*, arXiv preprint arXiv:1704.04932, (2017).

[5] M. Courbariaux, Y. Bengio, and J. David, *Binaryconnect: Training deep neural networks with binary weights during propagations*, In Advances in Neural Information Processing Systems, (2015), pp. 3123–3131.

[6] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, *Imagenet: A large-scale hierarchical image database*, Computer Vision and Pattern Recognition, IEEE Conference on, (2009).

[7] D. Donoho, *De-noising by soft-thresholding*, IEEE Trans. Info. Theory, 41 (1995), pp. 613–627.

[8] J. Fan and R. Li, *Variable selection via nonconcave penalized likelihood and its oracle properties*, J. Amer. Stat. Assoc., 96 (2001), pp. 1348–1360.

[9] T. Goldstein and S. Osher, *The split bregman method for ℓ_1-regularized problems*, SIAM J. Imaging Sci., 2 (2009), pp. 323–343.

[10] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, arXiv preprint arXiv:1512.03385, (2015).
[11] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, *Binarized neural networks: Training neural networks with weights and activations constrained to +1 or -1*, CoRR, (2016).
[12] S. Ioffe and C. Szegedy, *Normalization: Accelerating deep network training by reducing internal covariate shift*, arXiv preprint arXiv:1502.03167, (2015).
[13] A. Krizhevsky, *Learning multiple layers of features from tiny images*, (2009).
[14] A. Krizhevsky, I. Sutskever, and G. Hinton, *Imagenet classification with deep convolutional neural networks*, In Advances in neural information processing systems, (2012), pp. 1097–1105.
[15] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature, 521 (2015), pp. 436–444.
[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, In Proceedings of the IEEE, 86 (1998), pp. 2278–2324.
[17] F. Li, B. Zhang, and B. Liu, *Ternary weight networks*, arXiv preprint arXiv:1605.04711, (2016).
[18] H. Li, S. De, Z. Xu, C. Studer, H. Samet, and T. Goldstein, *Training quantized nets: A deeper understanding*, Advances in Neural Information Processing Systems, (2017).
[19] B. Mordukhovich, *Variational analysis and generalized differentiation I: Basic theory*, Springer Science & Business Media, 2006.
[20] J.-J. Moreau, *Proximité et dualité dans un espace hilbertien*, Bulletin de la Société Mathématique de France, 93 (1965), pp. 273–299.
[21] E. Park, J. Ahn, and S. Yoo, *Weighted-entropy-based quantization for deep neural networks*, Computer Vision and Pattern Recognition, IEEE Conference on, (2017), pp. 5456–5464.
[22] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, *Automatic differentiation in pytorch*, (2017).
[23] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, *Xnor-net: Imagenet classification using binary convolutional neural networks*, arXiv preprint arXiv:1603.05279, (2016).
[24] S. Ren, K. He, R. Girshick, and J. Sun, *Faster r-cnn: Towards real-time object detection with region proposal networks*, In Advances in neural information processing systems, (2015), pp. 91–99.
[25] R. Rockafellar and R. Wets, *Variational analysis*, Springer Science & Business Media, 2009.
[26] S. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, arXiv preprint arXiv:1409.1556, (2014).
[27] R. Tibshirani, *Regression shrinkage and selection via the lasso*, J. Roy. Statist. Soc. Ser. B., 58 (1996), pp. 267–288.
[28] P. Yin, Y. Lou, Q. He, and J. Xin, *Minimization of ℓ_{1-2} for compressed sensing*, SIAM J. Sci. Comput., 37 (2015), pp. A536–A563.
[29] P. Yin, S. Zhang, Y. Qi, and J. Xin, *Quantization and training of low bit-width convolutional neural networks for object detection*, arXiv preprint arXiv:1612.06052, (2016).
[30] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, *Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing*, SIAM J. Imaging Sci., 1 (2010), pp. 143–168.
[31] A. Zhou, A. Yao, Y. Guo, L. Xu, and Y. Chen, *Incremental network quantization: Towards lossless cnns with low-precision weights*, International Conference on Learning Representations, (2017).
[32] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, *Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients*, arXiv preprint arXiv: 1606.06160, (2016).
[33] C. Zhu, S. Han, H. Miao, and W. Dally, *Trained ternary quantization*, arXiv preprint arXiv:1612.01064, (2016).

Appendix: Technical Proofs.

Proof of Proposition 2.1. Since x^*_λ is the global minimizer of (10),

$$f^*_{\mathcal{Q}} \geq f(x^*_\lambda) + \frac{\lambda}{2}\mathrm{dist}(x^*_\lambda, \mathcal{Q})^2 \geq f^* + \frac{\lambda}{2}\mathrm{dist}(x^*_\lambda, \mathcal{Q})^2,$$

where $f^* = \min_{x\in\mathbb{R}^n} f(x) > -\infty$. So

$$\operatorname{dist}(x^*_\lambda, \mathcal{Q}) \le \sqrt{\frac{2(f^*_{\mathcal{Q}} - f^*)}{\lambda}} \to 0, \text{ as } \lambda \to \infty.$$

Denote $x^*_{\lambda,\mathcal{Q}} = \operatorname{proj}_{\mathcal{Q}}(x^*_\lambda)$, then $\|x^*_{\lambda,\mathcal{Q}} - x^*_\lambda\| \to 0$ as $\lambda \to \infty$. Since $f^*_{\mathcal{Q}}$ is the minimum in $\mathcal{Q}$, further we have

$$f(x^*_\lambda) + \frac{\lambda}{2}\operatorname{dist}(x^*_\lambda, \mathcal{Q})^2 \le f^*_{\mathcal{Q}} \le f(x^*_{\lambda,\mathcal{Q}}) \to f(x^*_\lambda), \text{ as } \lambda \to \infty.$$

Therefore, $\lim_{\lambda\to\infty} f(x^*_\lambda) = f^*_{\mathcal{Q}}$. ∎

Proof of Proposition 2.3. Proof by contradiction. Let us assume $x^* \in \mathcal{Q}$ is a local minimizer of problem (10) and $\nabla f(x^*) \neq \mathbf{0}$. Then for any point x in the neighborhood of x^*, we have

$$f(x^*) \le f(x) + \frac{\lambda}{2}\operatorname{dist}(x, \mathcal{Q})^2 \le f(x) + \frac{\lambda}{2}\|x - x^*\|^2.$$

Set $x = x^* - \beta\nabla f(x^*)$ with a small $\beta > 0$. The above inequality reduces to

$$f(x^*) \le f(x^* - \beta\nabla f(x^*)) + \frac{\lambda\beta^2}{2}\|\nabla f(x^*)\|^2. \tag{15}$$

On the other hand, by Taylor's expansion,

$$f(x^* - \beta\nabla f(x^*)) = f(x^*) - \beta\|\nabla f(x^*)\|^2 + o(\beta). \tag{16}$$

Combining (15) and (16), we have

$$\|\nabla f(x^*)\|^2 \le \frac{\lambda\beta}{2}\|\nabla f(x^*)\|^2 + o(1),$$

which leads to a contradiction as we let $\beta \to 0$. ∎

Proof of Proposition 2.2. Problem (11) is the same as

$$\min_x \min_{z\in\mathcal{Q}} \frac{1}{2}\|x - y^k\|^2 + \frac{\lambda}{2}\|z - x\|^2 = \min_{z\in\mathcal{Q}} \min_x \frac{1}{2}\|x - y^k\|^2 + \frac{\lambda}{2}\|z - x\|^2.$$

With fixed $z \in \mathcal{Q}$, the inner problem is minimized at $x = \frac{\lambda z + y^k}{\lambda+1}$. Then it reduces to

$$\begin{aligned} z^* &= \arg\min_{z\in\mathcal{Q}} \frac{1}{2}\left\|\frac{\lambda z + y^k}{\lambda+1} - y^k\right\|^2 + \frac{\lambda}{2}\left\|z - \frac{\lambda z + y^k}{\lambda+1}\right\|^2 \\ &= \arg\min_{z\in\mathcal{Q}} \|z - y^k\|^2 = \operatorname{proj}_{\mathcal{Q}}(y^k). \end{aligned}$$

Therefore, $x^k = \frac{\lambda \operatorname{proj}_{\mathcal{Q}}(y^k) + y^k}{\lambda+1}$ is the optimal solution. ∎

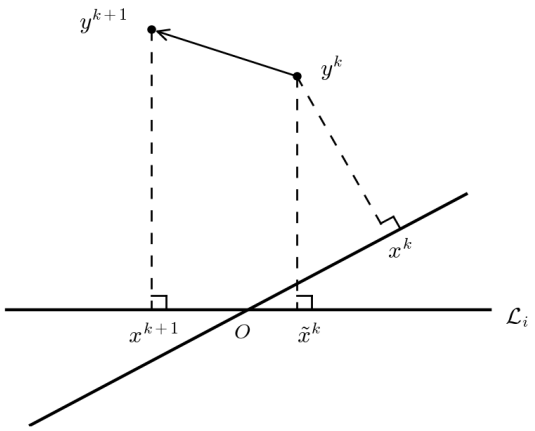

Figure 3. *Illustration of Lemma 4.6.* $y^{k+1} = y^k - \gamma_k \nabla f_k(x^k)$, *so* x^{k+1} *is also the projection of* $\tilde{x}^k - \gamma_k \nabla f_k(x^k)$ *onto* $\mathcal{L}_i$.

Proof of Lemma 4.4. Since $x^k, x^{k+1} \in \mathcal{Q}$ and $x^k = \mathrm{proj}_{\mathcal{Q}}(y^k)$, it holds that $\|y^k - x^k\|^2 \leq \|y^k - x^{k+1}\|^2$, i.e., $\alpha_k \geq 0$. ∎

Proof of Proposition 4.5. Since the only intersection of the line subspaces is the origin, the distance between x^k and any other line is at least $\|x^k\| \sin\theta_{\min}$. If $\|x^{k+1} - x^k\| < \|x^k\| \sin\theta_{\min}$, then x^k and x^{k+1} must lie in the same line, and therefore $\alpha_k = 1$. On the other hand, if α_k can only be 0, then it must hold that $\|x^k - y^k\| = \|x^k - y^{k+1}\|$ and $x^k \neq x^{k+1}$, meaning that x^{k+1} is a different projection of y^k onto $\mathcal{Q}$. Moreover, since the projection of $y^{k+1} = y^k - \gamma_k \nabla f_k(x^k)$ onto $\mathcal{Q}$ is also x^{k+1}. Suppose $x^{k+1} \in \mathcal{L}_i \subset \mathcal{Q}$, then $\nabla f_k(x^k) \perp \mathcal{L}_i$. ∎

Proof of Lemma 4.6. By the assumption, we have

$$x^{k+1} = \mathrm{proj}_{\mathcal{L}_i}(y^k - \gamma_k \nabla f_k(x^k)) = \mathrm{proj}_{\mathcal{L}_i}(\tilde{x}^k - \gamma_k \nabla f_k(x^k) + y^k - \tilde{x}^k).$$

Note that $y^k - \tilde{x}^k \perp \mathcal{L}_i$ (see Fig. 3), then

$$x^{k+1} = \mathrm{proj}_{\mathcal{L}_i}(\tilde{x}^k - \gamma_k \nabla f_k(x^k)).$$

So x^{k+1} is the closest point to $\tilde{x}^k - \gamma_k \nabla f_k(x^k)$ on $\mathcal{L}_i$. If $\tilde{x}^k - \gamma_k \nabla f_k(x^k) = \mathbf{0}$, then $x^{k+1} = \mathbf{0}$ is the global minimizer of (14). Otherwise, $x^{k+1} \neq \mathbf{0}$. Since the line subspaces that constitute $\mathcal{Q}$ only intersect at the origin, there exists a neighborhood $\mathcal{N}$ of x^{k+1} such that $\mathcal{N} \cap \mathcal{Q} \subset \mathcal{L}_i$. Therefore, x^{k+1} is a local minimizer of problem (14). ∎

Proof of Lemma 4.7. Using the facts $x^k = \mathrm{proj}_{\mathcal{Q}}(y^k)$, $\tilde{x}^k = \mathrm{proj}_{\mathcal{L}_i}(y^k) \in \mathcal{Q}$ and $x^{k+1} \in \mathcal{L}_i$ and invoking Lemma 4.4, we have

$$\begin{aligned} &\|x^{k+1} - \tilde{x}^k\|^2 = \|y^k - x^{k+1}\|^2 - \|y^k - \tilde{x}^k\|^2 \\ \leq &\|y^k - x^{k+1}\|^2 - \|y^k - x^k\|^2 = \alpha_k \|x^{k+1} - x^k\|^2. \end{aligned}$$

∎

Proof of Theorem 4.10. By Lemma 4.8,

$$\begin{aligned} f(x^{k+1}) &\le f(x^k) + \langle \nabla f(x^k), x^{k+1} - x^k \rangle + \frac{L}{2}\|x^{k+1} - x^k\|^2 \\ &= f(x^k) + \langle \nabla f_k(x^k), x^{k+1} - x^k \rangle + \langle \nabla f(x^k) - \nabla f_k(x^k), x^{k+1} - x^k \rangle + \frac{L}{2}\|x^{k+1} - x^k\|^2. \end{aligned} \tag{17}$$

The cross terms need care. We rewrite the update $x^{k+1} = \mathrm{proj}_{\mathcal{Q}}(y^k - \gamma_k \nabla f_k(x^k))$ as

$$x^{k+1} = \arg\min_{x \in \mathcal{Q}} \langle \nabla f_k(x^k), x \rangle + \frac{1}{2\gamma_k}\|x - y^k\|^2.$$

Since $x^k \in \mathcal{Q}$, we have

$$\langle \nabla f_k(x^k), x^{k+1} \rangle + \frac{1}{2\gamma_k}\|x^{k+1} - y^k\|^2 \le \langle \nabla f_k(x^k), x^k \rangle + \frac{1}{2\gamma_k}\|x^k - y^k\|^2.$$

Then by Lemma 4.4,

$$\langle \nabla f_k(x^k), x^{k+1} - x^k \rangle \le \frac{1}{2\gamma_k}(\|x^k - y^k\|^2 - \|x^{k+1} - y^k\|^2) \le -\frac{\underline{\alpha}}{2\gamma_k}\|x^{k+1} - x^k\|^2. \tag{18}$$

By Young's inequality,

$$\langle \nabla f(x^k) - \nabla f_k(x^k), x^{k+1} - x^k \rangle \le \frac{\gamma_k}{\underline{\alpha}}\|f(x^k) - \nabla f_k(x^k)\|^2 + \frac{\underline{\alpha}}{4\gamma_k}\|x^{k+1} - x^k\|^2. \tag{19}$$

Combining (17), (18) and (19) and taking the expectation gives

$$\mathbb{E}[f(x^{k+1})] \le \mathbb{E}[f(x^k)] - \frac{\underline{\alpha} - 2\gamma_k L}{4\gamma_k}\mathbb{E}[\|x^{k+1} - x^k\|^2] + \frac{\gamma_k \sigma^2}{\underline{\alpha}}. \tag{20}$$

Multiplying (20) by γ_k and using $\alpha_k \ge \underline{\alpha} > 0$, $\gamma_{k+1} \le \gamma_k \le \gamma < \frac{\underline{\alpha}}{2L}$ and $f \ge 0$, we obtain

$$\gamma_{k+1} E[f(x^{k+1})] \le \gamma_k \mathbb{E}[f(x^{k+1})] \le \gamma_k \mathbb{E}[f(x^k)] - (\underline{\alpha} - 2\gamma L)\mathbb{E}[\|x^{k+1} - x^k\|^2] + \frac{\gamma_k^2 \sigma^2}{\underline{\alpha}}$$

Rearranging terms in the above inequality and taking the sum over k, we have

$$(\underline{\alpha} - 2\gamma L)\sum_{k=0}^{\infty}\mathbb{E}[\|x^{k+1} - x^k\|^2] \le \gamma f(x^0) - \lim_{k\to\infty}\gamma_k \mathbb{E}[f(x^k)] + \frac{\sigma^2}{\underline{\alpha}}\sum_{k=0}^{\infty}\gamma_k^2 < \infty.$$

Therefore, $\lim_{k\to\infty}\mathbb{E}[\|x^{k+1} - x^k\|^2] = 0$.

Next we prove the second claim. By Lemma 4.6, the first-order optimality condition of (14) holds at x^{k+1}. So

$$\mathbf{0} \in \nabla f_k(x^k) + \frac{x^{k+1} - \tilde{x}^k}{\gamma_k} + \partial \chi_{\mathcal{Q}}(x^{k+1}),$$

which implies

$$-\frac{x^{k+1}-\tilde{x}^k}{\gamma_k}-\nabla f_k(x^k)+\nabla f(x^{k+1})\in\nabla f(x^{k+1})+\partial\chi_{\mathcal{Q}}(x^{k+1})=\partial h(x^{k+1}).$$

Therefore,

$$\begin{aligned}
&\mathbb{E}[\operatorname{dist}(\mathbf{0},\partial h(x^{k+1}))^2]\\
\leq&\mathbb{E}\left[\left\|-\frac{x^{k+1}-\tilde{x}^k}{\gamma_k}-\nabla f_k(x^k)+\nabla f(x^{k+1})\right\|^2\right]\\
\leq&\frac{1}{3}\left(\mathbb{E}[\left[\frac{\|x^{k+1}-\tilde{x}^k\|^2]}{\gamma_k^2}\right]+\mathbb{E}[\|\nabla f_k(x^k)-\nabla f(x^k)\|^2]+\mathbb{E}[\|\nabla f(x^k)-\nabla f(x^{k+1})\|^2]\right)\\
\leq&\frac{1}{3}\left(\bar{\alpha}\mathbb{E}\left[\frac{\|x^{k+1}-x^k\|^2}{\gamma_k^2}\right]+\sigma^2+L^2\mathbb{E}[\|x^{k+1}-x^k\|^2]\right). \qquad (21)
\end{aligned}$$

The second inequality above holds because of Cauchy-Schwarz inequality. In the last inequality, we used Lemma 4.7 and the assumption that f is L-Lipschitz differentiable. We want to bound $\liminf_{k\to\infty}\mathbb{E}\left[\frac{\|x^{k+1}-x^k\|^2}{\gamma_k^2}\right]$. From (20) it follows that

$$\gamma_k\left((\underline{\alpha}-2\gamma_k L)\mathbb{E}\left[\frac{\|x^{k+1}-x^k\|^2}{4\gamma_k^2}\right]-\frac{\sigma^2}{\underline{\alpha}}\right)\leq\mathbb{E}[f(x^k)-f(x^{k+1})].$$

Summing the above inequality over k yields

$$\sum_{k=0}^{\infty}\gamma_k\left((\underline{\alpha}-2\gamma_k L)\mathbb{E}\left[\frac{\|x^{k+1}-x^k\|^2]}{4\gamma_k^2}\right]-\frac{\sigma^2}{\underline{\alpha}}\right)\leq f(x^0)<\infty.$$

Since $\gamma_k>0$ and $\sum_{k=1}^{\infty}\gamma_k=\infty$, we must have

$$\liminf_{k\to\infty}\ (\underline{\alpha}-2\gamma_k L)\mathbb{E}\left[\frac{\|x^{k+1}-x^k\|^2]}{4\gamma_k^2}\right]-\frac{\sigma^2}{\underline{\alpha}}\leq 0.$$

and thus

$$\liminf_{k\to\infty}\ \mathbb{E}\left[\frac{\|x^{k+1}-x^k\|^2}{\gamma_k^2}\right]\leq\lim_{k\to\infty}\frac{4\sigma^2}{\underline{\alpha}(\underline{\alpha}-2\gamma_k L)}=\frac{4\sigma^2}{\underline{\alpha}^2}.$$

Finally, from (21) it follows that

$$\liminf_{k\to\infty}\ \mathbb{E}[\operatorname{dist}(\mathbf{0},\partial h(x^k))^2]\leq\frac{\sigma^2}{3}\left(\frac{4\bar{\alpha}}{\underline{\alpha}^2}+1\right),$$

which completes the proof. ∎