Supplementary Materials of Quantum-Classical Hybrid Quantized Neural Network

Wenxin Li[1]*, Chuan Wang[2]†, Hongdong Zhu[1], Qi Gao[1], Yin Ma[1], Hai Wei[1], Kai Wen[1]‡
[1]Beijing QBoson Quantum Technology Co., Ltd., Beijing 100015, China
[2]School of Artificial Intelligence, Beijing Normal University, Beijing 100875, China

May 22, 2025

1 Additional Related Work

In addition to the aforementioned work, there are also some other works related to quantum computing and neural networks. [1] introduces a novel training approach employing Adiabatic Quantum Computing (AQC), leveraging adiabatic evolution principles to address optimization problems. This universal AQC method proposed in [1] is designed for implementation on gate quantum computers. A hybrid approach is proposed in [2], which combines quantum and classical methods for accelerated training of BNN. The quantum portion utilizes the HHL algorithm to solve linear systems of equations for linear regression in a single layer BNN. In [3], the primary contribution lies in demonstrating the transfer of a trained artificial neural network to a quantum computing setting. The authors design a quadratic binary model for quantum annealing, aligning it with the behavior of a classical neural network by combining deep-learned parameters and layer structures.

2 Error Bound of Piecewise Linear Approximation in FNN

Proof: Define layer-wise activations:

$$h_0 = x, \quad h_l = \sigma(\mathbf{W}_l h_{l-1} + \mathbf{b}_l), \quad \hat{h}_0 = x, \quad \hat{h}_l = \hat{\sigma}(\mathbf{W}_l \hat{h}_{l-1} + \mathbf{b}_l), \quad l = 1, \ldots, L-1. \tag{1}$$

The output error is:

$$\|f(x) - \hat{f}(x)\|_2 = \|\mathbf{W}_L(h_{L-1} - \hat{h}_{L-1})\|_2 \leq \|h_{L-1} - \hat{h}_{L-1}\|_2, \tag{2}$$

since $\|\mathbf{W}_L\|_2 \leq 1$. Let $\delta_l = \|h_l - \hat{h}_l\|_2$. We bound δ_l recursively.

For layer l, consider:

$$h_l - \hat{h}_l = \sigma(\mathbf{W}_l h_{l-1} + \mathbf{b}_l) - \hat{\sigma}(\mathbf{W}_l \hat{h}_{l-1} + \mathbf{b}_l). \tag{3}$$

Decompose as:

$$h_l - \hat{h}_l = \left[\sigma(\mathbf{W}_l h_{l-1} + \mathbf{b}_l) - \sigma(\mathbf{W}_l \hat{h}_{l-1} + \mathbf{b}_l)\right] + \left[\sigma(\mathbf{W}_l \hat{h}_{l-1} + \mathbf{b}_l) - \hat{\sigma}(\mathbf{W}_l \hat{h}_{l-1} + \mathbf{b}_l)\right]. \tag{4}$$

The first term, using $L_\sigma \leq 1$, is:

$$\|\sigma(\mathbf{W}_l h_{l-1} + \mathbf{b}_l) - \sigma(\mathbf{W}_l \hat{h}_{l-1} + \mathbf{b}_l)\|_2 \leq L_\sigma \|\mathbf{W}_l h_{l-1} - \mathbf{W}_l \hat{h}_{l-1}\|_2 \leq \|\mathbf{W}_l\|_2 \delta_{l-1} \leq \delta_{l-1}. \tag{5}$$

*Email: liwx@boseq.com
†Corresponding Author. Email: wangchuan@bnu.edu.cn
‡Corresponding Author. Email: wenk@boseq.com

The second term, by the approximation error, satisfies:

$$\|\sigma(\mathbf{W}_l\hat{h}_{l-1}+\mathbf{b}_l)-\hat{\sigma}(\mathbf{W}_l\hat{h}_{l-1}+\mathbf{b}_l)\|_2=\sqrt{\sum_{i=1}^{m_l}|\sigma(z_i)-\hat{\sigma}(z_i)|^2}\leq\sqrt{m_l}\epsilon_\sigma\leq\sqrt{m}\epsilon_\sigma, \tag{6}$$

where $z=\mathbf{W}_l\hat{h}_{l-1}+\mathbf{b}_l$. Thus:

$$\delta_l\leq\delta_{l-1}+\sqrt{m}\epsilon_\sigma. \tag{7}$$

Solving the recursion, we can obtain that $\delta_l\leq l\sqrt{m}\epsilon_\sigma$. For the output:

$$\|f(x)-\hat{f}(x)\|_2\leq\delta_{L-1}\leq(L-1)\sqrt{m}\epsilon_\sigma\leq L\sqrt{m}\epsilon_\sigma. \tag{8}$$

To determine n, assume σ has $\|\sigma''\|_\infty\leq M$. Construct $\hat{\sigma}$ over $[-R,R]$ with n equal segments, each of width $h=\frac{2R}{n}$. The interpolation error is:

$$|\sigma(z)-\hat{\sigma}(z)|\leq\frac{1}{8}h^2\|\sigma''\|_\infty=\frac{1}{8}\left(\frac{2R}{n}\right)^2M=\frac{R^2M}{2n^2}. \tag{9}$$

Set $\frac{R^2M}{2n^2}\leq\epsilon$, we have

$$n\geq\sqrt{\frac{R^2M}{2\epsilon}},\quad n=O\left(\frac{1}{\sqrt{\epsilon}}\right). \tag{10}$$

□

3 Spline Quantumization Protocol

Quantum computing faces the formidable challenge of not only optimizing QUBO models but also addressing highly nonlinear optimization problems as optimization targets. The significance of this issue lies in its ubiquity across diverse scientific and industrial domains, where real-world problems often exhibit intricate, arbitrary relationships among variables. This fact underscores the urgency for quantum algorithms capable of navigating and optimizing complex, highly nonlinear landscapes.

Formally we consider the following problem:

$$\min_{\mathbf{x}\in\{0,1\}^n}f(h(\mathbf{x})), \tag{11}$$

where $f(\cdot)$ is a function of arbitrary form. This formal definition of the optimization problem unlockes the further potential of quantum computing in solving real-world optimization problems. A powerful approach to approximate highly nonlinear functions is to use spline interpolation. Splines are piecewise linear functions that provide a smooth and flexible fit to data points. Spline interpolation can transform the optimization landscape into a more tractable form, making it amenable to quantum optimization algorithms. The spline fitting process involves the following steps:

- **Knot Selection.** Choose a set of knots $\{M_i\}_{i=1}^n$ which are the points in the domain of $f(\cdot)$ where the piecewise polynomial segments will join. These knots should be chosen to adequately capture the variability of $f(\cdot)$ over the entire domain, as the placement of knots can significantly affect the accuracy of the spline approximation.
- **Spline Construction.** Define linear functions $S_i(x)$ for each segment between consecutive knots.
- **Spline Approximation.** Replace the original function $f(\cdot)$ with the spline function $S(\cdot)$. The spline function can be expressed as:

$$S(h(\mathbf{x}))=\sum_{i=1}^{n}S_i(h(\mathbf{x}))\cdot\beta_i, \tag{12}$$

  where $\beta_i\in\{0,1\}$ denotes whether $h(\mathbf{x})$ is in the i-th iterval.

3.1 Piecewise Constant Segment

An effective method for simplifying highly nonlinear functions is piecewise constant fitting. This approach approximates the function $f(\cdot)$ using constant segments, which can reduce the complexity of the optimization problem and facilitate the translation into a QUBO model.

$$\min \sum_{i=0}^{n} \beta_i \cdot S(M_i) \tag{13}$$

$$s.t \sum_{i=1}^{n} \beta_i \cdot M_{i-1} \leq h(\mathbf{x}) \leq \sum_{i=1}^{n} \beta_i \cdot M_i \tag{14}$$

$$\sum_{i=1}^{n} \beta_i = 1, \beta_i \in \{0,1\} \tag{15}$$

Dealing with the inequality constraint. We have now derived a QUBO model capable of approximately solving optimization problems with arbitrary objective functions in (3). However, it is important to note that in representing β_i, we introduced two inequality constraints. Here, we introduce the following two approaches:

- Adding a penalty term $(\sum_{i=1}^{n} \beta_i M_{i-1} + s - h(\mathbf{x}))^2$, where $s \in [0, \Delta M]$, assuming all the intervals are of length ΔM.
- For $s \in [-\frac{1}{2}, \frac{1}{2}]$, adding a penalty term

$$\sum_{i=1}^{n-1} \beta_i \cdot \Big(\frac{2h(\mathbf{x}) - (M_{i-1} + M_i)}{2(M_i - M_{i-1})} + s\Big)^2 + \Big(1 - \sum_{i=1}^{n-1} \beta_i\Big)\Big(\frac{2h(\mathbf{x}) - (M_{n-1} + M_n)}{2(M_n - M_{n-1})} + s\Big)^2. \tag{16}$$

It is important to note that the higher order terms cancel out when the expression is expanded.

We have the following theorem that elucidates the correctness of the second approach.

Theorem 1 *Let β^* be the optimal solution vector for the following optimization problem,*

$$\min_{\beta} \Big\{ \sum_{i=1}^{n} \beta_i \cdot \Big(\frac{2h(\mathbf{x}) - (M_{i-1} + M_i)}{2(M_i - M_{i-1})}\Big)^2 \Big| \sum_{i=1}^{n} \beta_i = 1 \Big\}. \tag{17}$$

Then $\beta_i^* = 1$ only at index $i = i_{\mathbf{x}}$, where $M_{i_{\mathbf{x}}-1} \leq h(\mathbf{x}) \leq M_{i_{\mathbf{x}}}$, while all other β_i^* are equal to 0.

Proof: To establish this conclusion, it suffices to observe the following: For $i = i_{\mathbf{x}}$, $|2h(\mathbf{x}) - (M_{i_{\mathbf{x}}-1} + M_{i_{\mathbf{x}}})| \leq M_{i_{\mathbf{x}}} - M_{i_{\mathbf{x}}-1}$ and hence $(\frac{2h(\mathbf{x}) - (M_{i_{\mathbf{x}}-1} + M_{i_{\mathbf{x}}})}{2(M_{i_{\mathbf{x}}} - M_{i_{\mathbf{x}}-1})})^2 \leq \frac{1}{4}$. And for $i \neq i_{\mathbf{x}}$, we have $(\frac{2h(\mathbf{x}) - (M_{i-1} + M_i)}{2(M_i - M_{i-1})})^2 > \frac{1}{4}$. □

Theorem 1 establishes that the penalty term in the new model effectively guides the values of β_i to accurately represent the interval in which $h(\mathbf{x})$ resides. In fact, this new penalty term represents the squared weighted sum of distances from $h(\mathbf{x})$ to the midpoints of each interval. Consequently, the minimization of this penalty term incentivizes β_i to take values that precisely indicate the correct interval for $h(\mathbf{x})$.

Building upon the versatility of spline approximation in simplifying complex optimization landscapes, we use the following sign function as an example to illustrates the efficiency of our approach.

Example 1 (Sign function) *sign(x) (1 if $x \geq 0$ else -1) can be represented as*

$$\text{sign}(x) = \min_{\beta \in \{0,1\}} \Big\{ \Big(2\beta - 1\Big) + \lambda \cdot \Big[\beta \cdot \Big(\frac{2x - M}{2M} + s\Big)^2 + (1 - \beta) \cdot (\frac{2x + M}{2M} + s)^2\Big] \Big\}, \tag{18}$$

where $M = \sup |x|$ and $s \in [-\frac{1}{2}, \frac{1}{2}]$ is a slack variable.

Compared with the model in [4], we do not need to introduce the intermediate variable. Additionally, when representing the sign activation function, we do not need to introduce auxiliary variables for order reduction, as done in [4]. These optimizations can save $O(WDN \log W)$ bits for us, where W, D, N denote the network width, network depth and dataset size respectively. Indeed we are able to reduce the coefficients of the highest-order terms in the model's qubit count by 50%.

3.2 Piecewise Linear Segment

Piecewise linear fitting approximates the function $f(\cdot)$ using linear segments. This method can effectively capture the behavior of moderately nonlinear functions while maintaining a manageable level of complexity in the optimization problem. According to [5], by using binary variables, minimizing a piecewise linear functions $S(\cdot)$ can be represented in linear form:

$$\min_{\alpha_i} \sum_{i=0}^{n} \alpha_i \cdot S(M_i) \tag{19}$$

$$\begin{aligned} \text{s.t} & \sum_{i=0}^{n} \alpha_i = 1 \\ & \alpha_0 \leq \beta_1, \alpha_n \leq \beta_n \\ & \alpha_i \leq \beta_{i+1} + \beta_i \ (1 \leq i \leq n-1) \\ & \sum_{i=1}^{n} \beta_i = 1, \beta_i \in \{0,1\} \end{aligned} \tag{20}$$

where $\alpha_i \in [0,1]$ and slope of the piecewise linear function changes at $M_i (1 \leq i \leq n)$.

Note that in this formulation, when $M_{k-1} \leq x \leq M_k$ for some k, then there exists some number $\alpha_{k-1} \in [0,1]$ such that

$$x = \alpha_{k-1} M_{k-1} + (1 - \alpha_k) M_k. \tag{21}$$

Since $S(\cdot)$ is linear in $[M_{k-1}, M_k]$, thus

$$S(x) = \alpha_{k-1} S(M_{k-1}) + (1 - \alpha_k) S(M_k) \tag{22}$$

We remark that in the context of QUBO modeling, the introduction of quadratic terms in the objective function allows for simplification by eliminating the need to explicitly represent the variable α_i using binary encoding. This optimization is particularly advantageous as it significantly reduces the required number of bits. In addition, the linear form presented earlier in the formulation introduces a notable drawback: the absence of the decision variables as independent entities. Instead, the decisions are indirectly represented through the coefficient α_i, signifying its position within a specified range. Consequently, an additional relationship between the decision variables and α_i needs to be established.

To address these considerations, an alternative formulation is proposed below. This formulation not only leverages the inclusion of quadratic terms but also rectifies the issue of $\mathbf{x}$ indirect representation:

$$\min \sum_{i=1}^{n+1} \left(\beta_{i-1} \cdot \frac{h(\mathbf{x}) - M_{i-1}}{M_i - M_{i-1}} + \beta_i \cdot \frac{M_i - h(\mathbf{x})}{M_i - M_{i-1}} \right) \cdot f(M_{i-1}) \tag{23}$$

$$s.t \sum_{i=1}^{n} \beta_i \cdot M_{i-1} \leq h(\mathbf{x}) \leq \sum_{i=1}^{n} \beta_i \cdot M_i \tag{24}$$

$$\sum_{i=1}^{n} \beta_i = 1, \beta_i \in \{0,1\} \tag{25}$$

For notational convenience, we set $\beta_0 = \beta_{n+1} = 0$. In the aforementioned objective function, we leverage the property that if $h(\mathbf{x})$ resides within the interval $[M_{i-1}, M_i]$, we approximate the values of $f(\cdot)$ within this interval using line segments connecting points $(M_{i-1}, f(M_{i-1}))$ and $(M_i, f(M_i))$ in the two-dimensional coordinates. The value of the objective function at the location of $h(\mathbf{x})$ along this linear segment is determined by the corresponding linear interpolation. Mathematically, the linear function representing the approximation of $f(\cdot)$ within the interval $[M_{i-1}, M_i]$ is expressed as:

$$S(h(\mathbf{x})) = f(M_{i-1}) + \frac{f(M_i) - f(M_{i-1})}{M_i - M_{i-1}} \cdot (h(\mathbf{x}) - M_{i-1})$$
$$= \frac{M_i - h(\mathbf{x})}{M_i - M_{i-1}} \cdot f(M_{i-1}) + \frac{h(\mathbf{x}) - M_{i-1}}{M_i - M_{i-1}} \cdot f(M_i) \tag{26}$$

We introduce the weighting factor β_i before the term in (26) to indicate whether $h(\mathbf{x})$ lies within the i-th interval $[M_{i-1}, M_i]$. The overall objective function aggregates these weighted values over all intervals, summarizing the importance of each interval's contribution to the optimization model.

4 Proof of Convergence of QCGD with Random Quantum Oracle

Proof: Similar as [6], the algorithm utilizes an augmented Lagrangian function, defined as

$$\mathcal{Q}_\alpha(\mathbf{V}, \mathbf{z}) = \mathrm{Tr}(\mathbf{CV}) + \mathbf{z}^\top(\mathcal{L}\mathbf{V} - \mathbf{v}) + \frac{\alpha}{2}\|\mathcal{L}\mathbf{V} - \mathbf{v}\|^2 \quad \text{for } \mathbf{V} \in \Delta^p, \tag{27}$$

where $\mathbf{V}$ represents the primal variable, $\mathbf{z}$ is the dual variable, α is the penalty parameter, $\mathbf{C}$ is the cost matrix, $\mathcal{L}$ denotes the matrix for linear constraints, and $\mathbf{v}$ is the vector of constraint values.

The QCGD algorithm begins by initializing α, $\mathbf{V}$ and $\mathbf{z}$, and then iterates through primal and dual updates. In the primal step, the algorithm computes the gradient of the augmented Lagrangian with respect to $\mathbf{V}$ and finds a direction that minimizes the linearized loss. This step involves solving a standard QUBO problem to determine the update direction. The primal variable $\mathbf{V}$ is then updated by taking a step towards this direction. In the dual step, the gradient of the augmented Lagrangian with respect to $\mathbf{z}$ is computed, and $\mathbf{z}$ is updated using gradient ascent. Over iterations, the penalty parameter α is increased as $\alpha_i = \alpha_0\sqrt{\delta i + 1}$ to ensure that $\mathbf{V}$ converges to a feasible solution that satisfies the constraints.

We can obtain the following inequalities:

- The smoothness of $\mathcal{Q}_{\alpha_t}$:

$$\mathcal{Q}_{\alpha_t}(\mathbf{V}_{t+1}, \mathbf{z}_t) \le \mathcal{Q}_{\alpha_t}(\mathbf{V}_t, \mathbf{z}_t) + \gamma_t \cdot \delta \cdot \mathrm{Tr}(\mathbf{Q}^{(t)}_{\mathrm{QUBO}}(\mathbf{V}_\star - \mathbf{V}_t)) + O\Big(\alpha_t\gamma_t^2 + \frac{\xi_t\gamma_t}{\sqrt{t}}\Big). \tag{28}$$

- Definition of $\mathbf{Q}^{(t)}_{\mathrm{QUBO}}$:

$$\gamma_t \cdot \delta \cdot \mathrm{Tr}(\mathbf{Q}^{(t)}_{\mathrm{QUBO}}(\mathbf{V}_\star - \mathbf{V}_t)) \le \gamma_t \cdot \delta \cdot \mathrm{Tr}(\mathbf{CV}_\star) - \gamma_t \cdot \delta \cdot \mathcal{Q}_{\alpha_t}(\mathbf{V}_t, \mathbf{z}_t) - \gamma_t \cdot \delta \cdot \frac{\alpha_t}{2}\|\mathcal{L}\mathbf{V}_t - \mathbf{v}\|^2. \tag{29}$$

Combining (28) and (29), we have

$$\begin{aligned}&\mathcal{Q}_{\alpha_t}(\mathbf{V}_{t+1}, \mathbf{z}_t) - \mathrm{Tr}(\mathbf{CV}_\star)\\ \le&(1 - \gamma_t \cdot \delta)\Big(\mathcal{Q}_{\alpha_t}(\mathbf{V}_t, \mathbf{z}_t) - \mathrm{Tr}(\mathbf{CV}_\star)\Big) - \gamma_t \cdot \delta \cdot \frac{\alpha_t}{2}\|\mathcal{L}\mathbf{V}_t - \mathbf{v}\|^2 + O\Big(\alpha_t\gamma_t^2 + \frac{\xi_t\gamma_t}{\sqrt{t}}\Big)\end{aligned} \tag{30}$$

Note that

$$\mathcal{Q}_{\alpha_t}(\mathbf{V}_t, \mathbf{z}_t) = \mathcal{Q}_{\alpha_{t-1}}(\mathbf{V}_t, \mathbf{z}_t) + \frac{\alpha_t - \alpha_{t-1}}{2}\|\mathcal{L}\mathbf{V}_t - \mathbf{v}\|^2 \tag{31}$$

Hence

$$\begin{aligned}&\mathcal{Q}_{\alpha_t}(\mathbf{V}_{t+1}, \mathbf{z}_t) - \mathrm{Tr}(\mathbf{CV}_\star)\\ \le&(1 - \gamma_t \cdot \delta)\Big(\mathcal{Q}_{\alpha_{t-1}}(\mathbf{V}_t, \mathbf{z}_t) - \mathrm{Tr}(\mathbf{CV}_\star)\Big) + ((1 - \gamma_t \cdot \delta)(\alpha_t - \alpha_{t-1})/2 - \gamma_t \cdot \delta \cdot \frac{\alpha_t}{2})\|\mathcal{L}\mathbf{V}_t - \mathbf{v}\|^2 + O\Big(\alpha_t\gamma_t^2 + \frac{\xi_t\gamma_t}{\sqrt{t}}\Big)\end{aligned} \tag{32}$$

By choosing $\gamma_i = \frac{2}{\delta\cdot(i+1)}$ and $\alpha_i = \alpha_0\sqrt{\delta i+1}$, we claim that

$$(1-\gamma_t\cdot\delta)(\alpha_t-\alpha_{t-1}) - \frac{\gamma_t\alpha_t}{2}\cdot\delta \le 0. \tag{33}$$

To verify this, first observe that

$$1-\gamma_t\delta = \frac{t-1}{t+1}, \quad \alpha_t-\alpha_{t-1} = \alpha_0\frac{\delta}{\sqrt{\delta t+1}+\sqrt{\delta(t-1)+1}}. \tag{34}$$

The LHS becomes:

$$\frac{\alpha_0}{t+1}\left(\frac{(t-1)\delta}{\sqrt{\delta t+1}+\sqrt{\delta(t-1)+1}} - \sqrt{\delta t+1}\right) \tag{35}$$

$$=\frac{\alpha_0}{t+1}\left(\frac{-\delta-1-\sqrt{(\delta t+1)(\delta(t-1)+1)}}{\sqrt{\delta t+1}+\sqrt{\delta(t-1)+1}}\right) \le 0 \tag{36}$$

Similar as [6], we can obtain the following bound:

$$\mathcal{Q}_{\alpha_t}(\mathbf{V}_{t+1},\mathbf{z}_{t+1}) - \mathrm{Tr}(\mathbf{C}\mathbf{V}_\star) \le (1-\gamma_t\cdot\delta)\Big(\mathcal{Q}_{\alpha_{t-1}}(\mathbf{V}_t,\mathbf{z}_t) - \mathrm{Tr}(\mathbf{C}\mathbf{V}_\star)\Big) + O\Big(\alpha_t\gamma_t^2 + \frac{\xi_t\gamma_t}{\sqrt{t}}\Big). \tag{37}$$

Define the error $e_t = \mathcal{Q}_{\alpha_{t-1}}(\mathbf{V}_t,\mathbf{z}_t) - \mathrm{Tr}(\mathbf{C}\mathbf{V}_\star)$. The recursion becomes:

$$e_{t+1} \le (1-\gamma_t\delta)e_t + O\left(\alpha_t\gamma_t^2 + \frac{\xi_t\gamma_t}{\sqrt{t}}\right). \tag{38}$$

Note that the terms:

- $1-\gamma_t\delta = 1-\frac{2}{t+1} = \frac{t-1}{t+1}$.
- $\alpha_t\gamma_t^2 = \alpha_0\sqrt{\delta t+1}\cdot\frac{4}{\delta^2(t+1)^2} \sim \frac{4\alpha_0}{\delta^{3/2}}t^{-3/2}$.
- $\frac{\varepsilon\gamma_t}{\sqrt{t}} = \varepsilon\cdot\frac{2}{\delta(t+1)}\cdot\frac{1}{\sqrt{t}} \sim \frac{2\varepsilon}{\delta}t^{-3/2}$.
- Total perturbation: $O\left(\left(\frac{4\alpha_0}{\delta^{3/2}}+\frac{2\varepsilon}{\delta}\right)t^{-3/2}\right)$.

Iterate the inequality:

$$e_{t+1} \le \prod_{s=1}^{t}(1-\gamma_s\delta)e_1 + \sum_{s=1}^{t}\left[\prod_{k=s+1}^{t}(1-\gamma_k\delta)\right]O\left(\alpha_s\gamma_s^2+\frac{\xi_s\gamma_s}{\sqrt{s}}\right) \tag{39}$$

$$=O\left(\frac{\frac{4\alpha_0}{\delta^{3/2}}}{t^{1/2}}\right) + O\Big(\frac{2}{\delta t^2}\cdot\underbrace{\sum_{s=1}^{t}\xi_s s^{1/2}}_{S(t)}\Big). \tag{40}$$

For the additive error term,

$$\mathbb{E}[S(t)] \le \sum_{s=1}^{t}\varepsilon s^{1/2} = O(\varepsilon t^{3/2}). \tag{41}$$

According to Chebyshev's inequality,

$$\mathbb{P}(S(t) \ge \mathbb{E}[S(t)] + \varepsilon t^{3/2}) \le \frac{\mathrm{Var}[S(t)]}{\varepsilon^2 t^3} \le \frac{\max\mathrm{Var}[\xi_s]}{\varepsilon^2 t}. \tag{42}$$

Therefore

$$\mathbb{E}[e_t] = O\Big(\frac{\frac{4\alpha_0}{\delta^{3/2}} + \frac{2\varepsilon}{\delta}}{t^{1/2}}\Big) \tag{43}$$

and

$$\mathbb{P}\Big[e_t = O\Big(\frac{\frac{4\alpha_0}{\delta^{3/2}} + \frac{2\varepsilon}{\delta}}{t^{1/2}}\Big)\Big] \geq 1 - \frac{\max \mathrm{Var}[\xi_s]}{\varepsilon^2 t}. \tag{44}$$

Consequently,

$$\mathbb{P}[\lim_{t\to\infty} e_t = 0] = 1.$$

Similar to the proof in [6],

$$\mathcal{Q}_{\alpha_t}(\mathbf{V}_{T+1}, \mathbf{z}_{T+1}) \geq \mathrm{Tr}(\mathbf{C}\mathbf{V}_{T+1}) - \frac{D^2}{2\alpha_t}, \tag{45}$$

we then have

$$\text{Objective-gap}_T = \mathrm{Tr}(\mathbf{C}\mathbf{V}_{T+1}) - \mathrm{Tr}(\mathbf{C}\mathbf{V}_\star) = O\Big(\frac{(1+\varepsilon)}{\delta^{3/2}\sqrt{T}}\Big) \tag{46}$$

For the infeasibility bound, the proof is similar as [6], we can derive the following bound:

$$\text{Infeasibility}_T = \|\mathcal{L}\mathbf{V}_{T+1} - \mathbf{v}\| = O\Big(\frac{(1+\varepsilon)}{\delta^{3/2}\sqrt{T}}\Big). \tag{47}$$

□

5 Analysis of Total Error Due to Truncation

The energy function of the Ising model is given by:

$$E[\sigma|J_{ij}, h_i] = -\sum_{i<j} J_{ij}\sigma_i\sigma_j - \sum_i h_i\sigma_i. \tag{48}$$

The objective function of QUBO model is

$$f(\mathbf{x}|q_{ij}) = \sum_{i\neq j} q_{ij}x_ix_j + \sum_i q_ix_i. \tag{49}$$

Let σ^* be the optimal configurations under origin Ising coefficients J_{ij}s and h_is, $\mathbf{x}^*$ be the corresponding optimal solution to the QUBO model with origin QUBO coefficients q_{ij}s and q_is. We use $\sigma^{*\prime}$ to denote the optimal configurations under truncated Ising coefficients, and let $\mathbf{x}^{*\prime}$ be the corresponding solution in QUBO model. Then we have the following conclusion

Lemma 1 *If the Ising coefficients are truncated to d bits, the error due to truncation is in the order of*

$$\Delta \textit{Hamiltonian} = |E[\sigma|J_{ij}, h_i] - E[\sigma^{*\prime}|J_{ij}, h_i]| = O(2^{-d}\cdot n^2), \tag{50}$$

$$\Delta \textit{QUBO Value} = |f(\mathbf{x}|q_{ij}) - f(\mathbf{x}^{*\prime}|q_{ij})| = O(2^{-d}\cdot n^2). \tag{51}$$

Proof: The change in the Ising coefficients due to truncations is

$$|J_{ij} - J'_{ij}|, |h_i - h'_i| = O(2^{-d}), \tag{52}$$

consequently for any σ

$$|E[\sigma|J_{ij}, h_i] - E[\sigma|J'_{ij}, h'_i]| = O(2^{-d}\cdot n^2). \tag{53}$$

Note that

$$\begin{aligned} E[\sigma^{*\prime}|J_{ij}, h_i] \leq & E[\sigma^{*\prime}|J'_{ij}, h'_i] + O(2^{-d} \cdot n^2) \\ \leq & E[\sigma^{*}|J'_{ij}, h'_i] + O(2^{-d} \cdot n^2), \end{aligned} \tag{54}$$

and

$$\begin{aligned} E[\sigma^{*\prime}|J_{ij}, h_i] \geq & E[\sigma^{*}|J_{ij}, h_i] \\ \geq & E[\sigma^{*}|J'_{ij}, h'_i] - O(2^{-d} \cdot n^2), \end{aligned} \tag{55}$$

we can obtain (50) by combining (54) and (55).

To convert the Ising model to the QUBO model, we replace the spin variables σ_i with binary variables x_i, by using the following equation:

$$\sigma_i = 2x_i - 1 \tag{56}$$

Substitute this into the Ising model energy function:

$$E = -\sum_{i<j} J_{ij}(2x_i - 1)(2x_j - 1) - \sum_i h_i(2x_i - 1) \tag{57}$$

$$= -\sum_{i<j} J_{ij}\left[4x_i x_j - 2x_i - 2x_j + 1\right] - \sum_i h_i\left[2x_i - 1\right] \tag{58}$$

$$= -\sum_{i<j} 4J_{ij}x_i x_j + \sum_i (2\sum_{j\neq i} J_{ij} - 2h_i)x_i - \sum_{i<j} J_{ij} + \sum_i h_i. \tag{59}$$

Hence $E[\sigma|J_{ij}, h_i] = f(\mathbf{x}|q_{ij}) + (\sum_i h_i - \sum_{i<j} J_{ij})$ and

$$\Delta\text{QUBO Value} = \Delta\text{Hamiltonian} + O(2^{-d} \cdot n^2) = O(2^{-d} \cdot n^2). \tag{60}$$

When $d = \Omega(\log n)$, the additive error is in the order of $O(\frac{1}{\sqrt{t}})$ for all $t \leq T$, and thus the convergence follows from Proposition 1. □

6 Additional Details of Experimental Results

The training set consists of 12000 images (6000 coat and 6,000 sandal images), and the test set consists of 2000 images (1000 coat and 1000 sandal images). Each image is divided into 3 columns vertically, resulting in two 28×9 grid and one 28×10 grid. We then calculate the number of zero pixels in each column and use two threshold values to determine the values in a length-3 vector for each image. This vector represents the image as -1, 0, or 1, based on the comparison with the thresholds.

A neural network with a hidden layer of depth 1, width 2, and the *Sigmoid* activation function $\sigma(x) = \frac{1}{1+e^{-x}}$ is used for classification. We approximate the sigmoid activation function with a piecewise constant function. The breakpoints in this experiment is $\{-8, -4, 0, 4, 8\}$ and the value of the piecewise linear function in each interval is equal to the function value of sigmoid function at the midpoint of that interval.

7 Principles of STE and BinaryConnect

Straight-Through Estimator (STE) [7, 8]

The Straight-Through Estimator (STE) enables gradient-based training for non-differentiable operations, such as quantization, in neural networks. During the forward pass, a weight x_i is quantized as:

$$\hat{x}_i = s \cdot \text{clamp}\left(\left\lfloor \frac{x_i}{s} \right\rfloor, n, p\right), \tag{61}$$

where s is a scaling factor, clamp$(\cdot)$ ensures the quantized value lies within integer bounds $n = q_{\min}/s$ and $p = q_{\max}/s$, and the rounding operation $\lfloor \cdot \rceil$ is non-differentiable. STE approximates the gradient of the rounding operation as 1:

$$\frac{\partial \lfloor \omega \rceil}{\partial \omega} = 1. \tag{62}$$

This allows the gradient to "pass through" the non-differentiable operation, enabling backpropagation for quantized neural networks.

BinaryConnect [9]

BinaryConnect trains neural networks with binary weights, to reduce computational and memory costs. It maintains real-valued weights w during training, which are binarized during the forward and backward passes using the sign function:

$$w_b = \text{sign}(w), \tag{63}$$

where $w_b \in \{-1, +1\}$. In the forward pass, binary weights w_b are used to compute activations. In the backward pass, the gradient of the loss C with respect to the binary weights, $\frac{\partial C}{\partial w_b}$, is computed and used to update the real-valued weights. This allows BinaryConnect to accumulate gradient updates in the real-valued weights while operating with binary weights.

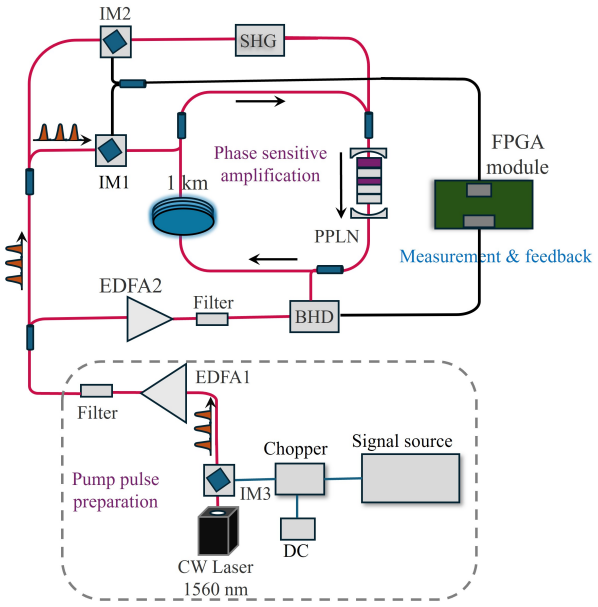

Figure 1: Schematic of the coherent Ising machine (CIM) structure, utilizing optical parametric oscillators and phase-sensitive amplification to solve optimization problems.

Coherent Ising Machine

As shown in Figure 1, we implement the CIM, an optical system engineered to address combinatorial optimization problems through optical parametric oscillators (OPOs). A continuous wave (CW) laser at 1560 nm initiates the process, feeding into a pump pulse preparation stage. Here, intensity modulators (IM1, IM2, IM3), erbium-doped fiber amplifiers (EDFA1, EDFA2), and filters shape and amplify the light into pump pulses. These pulses enter a second harmonic generation (SHG) unit, producing a frequency-doubled signal, which is then directed into a periodically poled lithium niobate (PPLN) crystal. The PPLN facilitates phase-sensitive amplification, forming a degenerate OPO that generates coherent signal and idler photons.

These photons are measured via balanced homodyne detection (BHD), with an FPGA module providing feedback to steer the system toward optimal solutions by mapping the problem onto the Ising model's energy landscape.

CIM operates by simulating the Ising Hamiltonian, defined as

$$H = -\sum_{i,j} J_{ij}\sigma_i\sigma_j - \sum_i h_i\sigma_i, \tag{64}$$

where $\sigma_i \in \{1, -1\}$ represents spins, J_{ij} denotes coupling strengths, and h_i accounts for external fields. In the PPLN crystal, pump pulses drive parametric amplification, creating optical pulses that encode the spins as two possible phase states. The BHD measures the pulses' phase and amplitude, determining the system's energy state. The FPGA uses this data to adjust parameters like pump intensity, effectively minimizing the Hamiltonian through feedback. A chopper and intensity modulators ensure stable pulse circulation in the loop, while the feedback introduces coupling between pulses, mimicking the $J_{ij}\sigma_i\sigma_j$ interaction terms.

The system's dynamics allow it to explore the Ising energy landscape efficiently. The optical pulses evolve in parallel, with their interactions governed by the feedback loop, driving the system toward the ground state of the Hamiltonian, which corresponds to the optimization problem's solution. The amplifiers and filters maintain pulse quality, while the PPLN's nonlinear efficiency ensures robust signal generation. The BHD's noise suppression enhances measurement precision, enabling the CIM to handle complex problems by leveraging the coherence and parallelism of optical signals.

References

[1] Abel, S., Criado, J. C. & Spannowsky, M. Training neural networks with universal adiabatic quantum computing. *arXiv:2308.13028* (2023).

[2] Alarcon, S. L., Merkel, C., Hoffnagle, M., Ly, S. & Pozas-Kerstjens, A. Accelerating the training of single layer binary neural networks using the hhl quantum algorithm. In *2022 IEEE 40th International Conference on Computer Design (ICCD)*, 427–433 (20).

[3] Higham, C. F. & Bedford, A. Quantum deep learning by sampling neural nets with a quantum annealer. *Scientific Reports* **13**, 3939 (2023).

[4] Song, X. *et al.* Training multi-layer neural networks on ising machine. *arXiv:2311.03408* (2023).

[5] Winston, W. L. *Operations research: applications and algorithms* (Cengage Learning, 2022).

[6] Yurtsever, A., Birdal, T. & Golyanik, V. Q-fw: A hybrid classical-quantum frank-wolfe for quadratic binary optimization. In *European Conference on Computer Vision (ECCV)*, 352–369 (2022).

[7] Bengio, Y., Léonard, N. & Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432* .

[8] Nagel, M. *et al.* A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295* .

[9] Courbariaux, M., Bengio, Y. & David, J.-P. Binaryconnect: Training deep neural networks with binary weights during propagations. *Advances in neural information processing systems* (2015).