Quantum algorithm for training nonlinear SVMs in almost linear time

Jonathan Allcock[1,*] and Chang-Yu Hsieh[1,†]

[1] *Tencent Quantum Laboratory*

(Dated: June 19, 2020)

We propose a quantum algorithm for training nonlinear support vector machines (SVM) for feature space learning where classical input data is encoded in the amplitudes of quantum states. Based on the classical algorithm of Joachims [1], our algorithm has a running time which scales linearly in the number of training examples (up to polylogarithmic factors) and applies to the standard soft-margin ℓ_1-SVM model. In contrast, the best classical algorithms have super-linear scaling for general feature maps, and achieve linear m scaling only for linear SVMs, where classification is performed in the original input data space, or for the special case of feature maps corresponding to shift-invariant kernels. Similarly, previously proposed quantum algorithms either have super-linear scaling in m, or else apply to different SVM models such as the hard-margin or least squares ℓ_2-SVM, which lack certain desirable properties of the soft-margin ℓ_1-SVM model. We classically simulate our algorithm and give evidence that it can perform well in practice, and not only for asymptotically large data sets.

I. INTRODUCTION

Support vector machines (SVMs) are powerful supervised learning models which perform classification by identifying a decision surface which separates data according to their labels [2, 3]. While classifiers based on deep neural networks have increased in popularity in recent years, SVM-based classifiers maintain a number of advantages which make them an appealing choice in certain situations. SVMs are simple models with a smaller number of trainable parameters than neural networks, and thus can be less prone to overfitting and easier to interpret. Furthermore, neural network training may often get stuck in local minima, whereas SVM training is guaranteed to find a global optimum [4]. For problems such as text classification which involve high dimensional but sparse data, linear SVMs — which seek a separating hyperplane in the same space as the input data — have been shown to perform extremely well, and training algorithms exist which scale efficiently, i.e. linearly, in the number of training examples m [5–7].

In more complex cases, where a nonlinear decision surface is required to classify the data suc-

* jonallcock@tencent.com

† kimhsieh@tencent.com

cessfully, nonlinear SVMs can be used, which seek a separating hyperplane in a higher dimensional feature space. Such feature space learning typically makes use of the kernel trick [8], a method enabling inner product computations in high or even infinite dimensional spaces to be performed implicitly, without requiring the explicit and resource intensive computation of the feature vectors themselves.

While powerful, the kernel trick comes at a cost: the classical training time of nonlinear SVMs scales poorly with m. Indeed, storing the kernel matrix in memory itself requires $O(m^2)$ resources, making subquadratic training times impossible in the worst case. In practice, advanced solvers employ multiple heuristics to improve their performance, which makes rigorous analyses of their performance difficult. However, methods like SVM-Light [9], SMO [10], LIBSVM [11] and SVM-Torch [12] still empirically scale approximately quadratically with m for nonlinear SVMs. Finding algorithms with better time dependence on m is therefore increasingly important, as such super-linear scaling becomes intractable for large scale datasets, ubitquitous in the modern era of big data.

One exception is the case of so-called shift-invariant kernels [13], which include the popular Gaussian radial basis function (RBF) kernel, where classical sampling techniques can be used to map the high dimensional data into a random low dimensional feature space, which can then trained by fast linear methods. This method has empirically competed favorably with more sophisticated kernel machines in terms of classification accuracy, at a fraction of the training time. While such a method seems to strike a balance between linear and nonlinear approaches, it cannot be applied to more general kernels.

Can quantum computers implement SVMs more effectively than classical computers? Rebentrost and Lloyd were the first to consider this question [14], and since then numerous other proposals have been put forward [15–19]. While the details vary, at a high level these quantum algorithms aim to bring benefits in two main areas: i) faster training and evaluation time of SVMs or ii) greater representational power by encoding the high dimensional feature vectors in the amplitudes of quantum states. Such quantum feature maps enable high dimensional inner products to be computed directly and, by sidestepping the kernel trick, allow classically intractable kernels to be computed. These proposals are certainly intriguing, and open up new possibilities for supervised learning. However, the proposals to date with improved running time dependence on m for nonlinear SVMs do not apply to the standard soft-margin ℓ_1-SVM model, but rather to variations such as least squares ℓ_2-SVMs[14] or hard-margin SVMs [19]. While these other models are useful in certain scenarios, soft-margin ℓ_1-SVMs have two properties - sparsity of weights and robustness to

noise - that make them preferable in many circumstances.

In this work we present a quantum method to train nonlinear soft-margin ℓ_1-SVMs with quantum feature maps in a time that scales linearly (up to polylogarithmic factors) in the number of training examples, and which is not restricted to shift-invariant kernels. Our approach is based on the elegant linear time classical algorithm of Joachims [1] for linear SVMs, adapted to make use of quantum computers for approximating the kernel matrix of a related learning model known as a structural SVM [20]. Provided that one has quantum access to the classical data, i.e. quantum random access memory (qRAM) [21, 22], quantum states corresponding to sums of feature vectors can be efficiently created, and then standard methods employed to approximate the inner products between such quantum states. As the output of the quantum procedure is only an approximation to a desired positive semi-definite (p.s.d.) matrix , it is not itself guaranteed to be p.s.d., and hence an additional classical projection step must be carried out to map on to the p.s.d. cone at each iteration.

Before stating our result in more detail, let us make one remark. It has recently been shown by Tang [23] that the data-structure required for efficient qRAM-based inner product estimation would also enable such inner products to be estimated classically, with only a polynomial slow-down relative to quantum, and her method has been employed to de-quantize a number of quantum machine learning algorithms [23–25] based on such data-structures. However, in practice, polynomial factors can make a difference, and an analysis of a number of such quantum-inspired classical algorithms [26] concludes that care is needed when assessing their performance relative to the quantum algorithms from which they were inspired. More importantly, in this current work, the quantum states produced using qRAM access are subsequently mapped onto a larger Hilbert space before their inner products are evaluated. This means that the procedure cannot be de-quantized in the same way.

II. BACKGROUND AND RESULTS

To state our results more precisely, it is first necessary to introduce classical soft-margin ℓ_1-SVMs and structural SVMs.

Let $S = \{(\mathbf{x}_1, y_1), \dots (\mathbf{x}_m, y_m)\}$ be a data set with $\mathbf{x}_i \in \mathbb{R}^d$, and labels $y_i \in \{+1, -1\}$. Let $\Phi : \mathbb{R}^d \to \mathcal{H}$ be a feature map where $\mathcal{H}$ is a real Hilbert space (of finite or infinite dimension) with inner product $\langle \cdot , \cdot \rangle$, and let $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be the associated kernel function defined by $K(\mathbf{x}, \mathbf{y}) \stackrel{\mathsf{def}}{=} \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$. Let $R = \max_i \|\Phi(\mathbf{x}_i)\|$ denote the largest ℓ_2 norm of the feature mapped

vectors. In what follows, $\|\cdot\|$ will always refer to the ℓ_2 norm, and other norms will be explicitly differentiated.

A. Support Vector Machine Training

Training a soft-margin ℓ_1-SVM with parameter $C > 0$ corresponds to solving the following optimization problem:

OP 1. *(SVM Primal)*

$$\begin{aligned} \min_{\mathbf{w}\in\mathcal{H},\ \xi_i\geq 0} \quad & \frac{1}{2}\langle \mathbf{w}, \mathbf{w}\rangle + \frac{C}{m}\sum_{i=1}^{m}\xi_i \\ s.t. \quad & y_i \langle \mathbf{w}, \Phi(x_i)\rangle \geq 1-\xi_i \qquad \forall i = 1,\ldots,m. \end{aligned}$$

Note that, following [1], we divide $\sum_i \xi_i$ by m to capture how C scales with the training set size. The trivial case $\Phi(\mathbf{x}) = \mathbf{x}$ corresponds to a linear SVM, i.e. a separating hyperplane is sought in the original input space. When one considers feature maps $\Phi(\mathbf{x})$ in a high dimensional space, it is more practical to consider the dual optimization problem, which is expressed in terms of inner products, and hence the kernel trick can be employed.

OP 2. *(SVM Dual)*

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\sum_{i,j=1}^{m} y_i\alpha_i y_j\alpha_j K(\mathbf{x}_i,\mathbf{x}_j) + \sum_{i=1}^{m}\alpha_i \\ s.t. \quad & 0 \leq \alpha_i \leq \frac{C}{m} \qquad \forall i = 1,\ldots,m \end{aligned}$$

This is a convex quadratic program with box constraints, for which many classical solvers are available, and which requires time polynomial in m to solve. For instance, using the barrier method [27] a solution can be found to within ε_b in time $O(m^4 \log(m/\varepsilon_b))$. Indeed, even the computation of the kernel matrix K takes time $\Theta(m^2)$, so obtaining subquadratic training times via direct evaluation of K is not possible.

B. Structural SVMs

Joachims [1] showed that an efficient approximation algorithm - with running time $O(m)$ - for linear SVMs could be obtained by considering a slightly different but related model known as a *structural SVM*, which makes use of linear combinations of label-weighted feature vectors:

Definition 1. *For a given data set* $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$*, feature map* Φ*, and* $\mathbf{c} \in \{0,1\}^m$*, define*

$$\Psi_{\mathbf{c}} \stackrel{\text{def}}{=} \frac{1}{m} \sum_{i=1}^{m} c_i y_i \Phi(\mathbf{x}_i)$$

With this notation, the structural SVM primal and dual optimization problems are:

OP 3. *(Structural SVM Primal)*

$$\begin{aligned} \min_{\mathbf{w} \in \mathcal{H},\ \xi \geq 0} \quad & P(\mathbf{w}, \xi) \stackrel{\text{def}}{=} \frac{1}{2} \langle \mathbf{w}, \mathbf{w} \rangle + C\xi \\ s.t. \quad & \frac{1}{m} \sum_{i=1}^{m} c_i - \langle \mathbf{w}, \Psi_{\mathbf{c}} \rangle \leq \xi, \qquad \forall \mathbf{c} \in \{0,1\}^m. \end{aligned}$$

OP 4. *(Structural SVM Dual)*

$$\begin{aligned} \max_{\alpha \geq 0} \quad & D(\alpha) \stackrel{\text{def}}{=} -\frac{1}{2} \sum_{\mathbf{c}, \mathbf{c}' \in \{0,1\}^m} \alpha_{\mathbf{c}} \alpha_{\mathbf{c}'} J_{\mathbf{c}\mathbf{c}'} + \sum_{\mathbf{c} \in \{0,1\}^m} \frac{\|\mathbf{c}\|_1}{m} \alpha_{\mathbf{c}} \\ s.t. \quad & \sum_{\mathbf{c} \in \{0,1\}^m} \alpha_{\mathbf{c}} \leq C \end{aligned}$$

where $J_{\mathbf{c}\mathbf{c}'} = \langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle$ *and* $\|\cdot\|_1$ *denotes the* ℓ_1*-norm.*

Whereas the original SVM problem OP 1 is defined by m constraints and m slack variables ξ_i, the structural SVM OP 3 has only one slack variable ξ but 2^m constraints, corresponding to each possible binary vector $\mathbf{c} \in \{0,1\}^m$. In spite of these differences, the solutions to the two problems are equivalent in the following sense.

Theorem 1 (Joachims [1])**.** *Let* $(\mathbf{w}^*, \xi_1^*, \ldots, \xi_m^*)$ *be an optimal solution of OP 1, and let* $\xi^* = \frac{1}{m} \sum_{i=1}^{m} \xi_i^*$*. Then* $(\mathbf{w}^*, \xi^*)$ *is an optimal solution of OP 3 with the same objective function value. Conversely, for any optimal solution* $(\mathbf{w}^*, \xi^*)$ *of OP 3, there is an optimal solution* $(\mathbf{w}^*, \xi_1^*, \ldots, \xi_m^*)$ *of OP 1 satisfying* $\xi^* = \frac{1}{m} \sum_{i=1}^{m} \xi_i$*, with the same objective function value.*

While elegant, Joachim's algorithm can achieve $O(m)$ scaling only for linear SVMs — as it requires explicitly computing a set of vectors $\{\Psi_{\mathbf{c}}\}$ and their inner products — or to shift-invariant kernels where sampling methods can be used to approximate sums of kernel evaluations. For high dimensional feature maps Φ not corresponding to shift invariant kernels, computing $\Psi_{\mathbf{c}}$ classically is inefficient. We propose instead to embed the feature mapped vectors $\Phi(\mathbf{x})$ and linear combinations $\Psi_{\mathbf{c}}$ in the amplitudes of quantum states, and compute the required inner products efficiently using a quantum computer.

C. Our Results

In Section III we will formally introduce the concept of a quantum feature map. For now it is sufficient to view this as a quantum circuit which, in time T_Φ, realizes a feature map $\Phi : \mathbb{R}^d \to \mathcal{H}$, with maximum norm $\max_{\mathbf{x}} \|\Phi(\mathbf{x})\| = R$, by mapping the classical data into the state of a multi-qubit system.

Our first main result is a quantum algorithm with running time linear in m that generates an approximately optimal solution for the structural SVM problem. By Theorem 1, this is equivalent to solving the original soft-margin ℓ_1-SVM.

Quantum nonlinear SVM training*: [See Theorems 6 and 7] There is a quantum algorithm that, with probability at least* $1-\delta$*, outputs* $\hat{\alpha}$ *and* $\hat{\xi}$ *such that if* $(\mathbf{w}^*, \xi^*)$ *is the optimal solution of OP 3, then*

$$P(\hat{\mathbf{w}}, \hat{\xi}) - P(\mathbf{w}^*, \xi^*) \leq \min\left\{\frac{C\epsilon}{2}, \frac{\epsilon^2}{8R^2}\right\}$$

where $\hat{\mathbf{w}} \stackrel{\mathsf{def}}{=} \sum_{\mathbf{c}} \hat{\alpha}_{\mathbf{c}} \Psi_{\mathbf{c}}$*, and* $(\hat{\mathbf{w}}, \hat{\xi} + 3\epsilon)$ *is feasible for OP 3. The running time is*

$$\tilde{O}\left(\frac{CR^3 \log(1/\delta)}{\Psi_{\min}}\left(\frac{t_{\max}^2}{\epsilon} \cdot m + t_{\max}^5\right) T_\Phi\right)$$

where $t_{\max} = \max\left\{\frac{4}{\epsilon}, \frac{16CR^2}{\epsilon^2}\right\}$*,* T_Φ *is the time required to compute feature map* Φ *on a quantum computer and* $\Psi_{\min}$ *is a term that depends on both the data as well as the choice of quantum feature map.*

Here and in what follows, the tilde big-O notation hides polylogarithmic terms. In the Simulation section we show that, in practice, the running time of the algorithm can be significantly faster than the theoretical upper-bound. The solution $\hat{\alpha}$ is a $t_{\max}$-sparse vector of total dimension 2^m. Once it has been found, a new data point $\mathbf{x}$ can be classified according to

$$\begin{aligned} y_{pred} &= \operatorname{sgn}\left\langle \sum_{\mathbf{c}} \hat{\alpha}_{\mathbf{c}} \Psi_{\mathbf{c}}, \Phi(\mathbf{x}) \right\rangle \\ &= \operatorname{sgn}\left(\sum_{\mathbf{c}} \hat{\alpha}_{\mathbf{c}} \sum_{i=1}^{m} \frac{c_i y_i}{m} \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}) \rangle\right) \end{aligned}$$

where y_{pred} is the predicted label of $\mathbf{x}$. This is a sum of $O(mt_{\max})$ inner products in feature space, which classical methods require time $O(mt_{\max})$ evaluate in general. Our second result is a quantum algorithm for carrying out this classification with running time independent of m.

Quantum nonlinear SVM classification*: [See Theorem 8] There is a quantum algorithm*

which, in time

$$\tilde{O}\left(\frac{CR^3\log(1/\delta)}{\Psi_{\min}}\frac{t_{\max}}{\epsilon}T_\Phi\right)$$

outputs, with probability at least $1-\delta$*, an estimate to* $\langle\sum_{\mathbf{c}}\hat{\alpha}_{\mathbf{c}}\Psi_{\mathbf{c}},\Phi(\mathbf{x})\rangle$ *to within* ϵ *accuracy.* The sign of the output is then taken as the predicted label.

III. METHODS

Our results are based on three main components: Joachim's linear time classical algorithm, quantum feature maps, and efficient quantum methods for estimating inner products of linear combinations of high dimensional vectors.

A. Joachim's linear time algorithm for linear SVMs

On the surface, the structural SVM problems OP 3 and OP 4 look more complicated to solve than the original SVM problems OP 1 and OP 2. However, it turns out that the solution α^* to OP 4 is highly sparse and, consequently, the structural SVM admits an efficient algorithm. Joachims's original procedure is presented in Algorithm 1.

Algorithm 1 Training structural SVMs via OP 3

Input: Training set $S=\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_m,y_m)\}$. SVM hyperparameter $C>0$. Tolerance $\epsilon>0$.

$\mathbf{c}\in\{0,1\}^m$.

$\mathcal{W}\leftarrow\{\mathbf{c}\}$

repeat

$(\mathbf{w},\xi)\leftarrow\arg\min_{\mathbf{w},\xi\geq 0}\frac{1}{2}\langle\mathbf{w},\mathbf{w}\rangle+C\xi$

s.t. $\forall\mathbf{c}\in\mathcal{W}:\quad\frac{1}{m}\sum_{i=1}^m c_i-\langle\mathbf{w},\Psi_{\mathbf{c}}\rangle\leq\xi$

for $i=1,\ldots,m$ **do**

$$c_i^*\leftarrow\begin{cases}1 & y_i\left(\mathbf{w}^T\mathbf{x}_i\right)<1\\ 0 & \text{otherwise}\end{cases}$$

$\mathcal{W}\leftarrow\mathcal{W}\cup\{\mathbf{c}^*\}$

until $\frac{1}{m}\sum_{i=1}^m c_i^*-\sum\alpha_{\mathbf{c}'\in\mathcal{W}}\langle\Psi_{\mathbf{c}'},\Psi_{\mathbf{c}^*}\rangle\leq\xi+\epsilon$

Output: $(\mathbf{w},\xi)$

The main idea behind Algorithm 1 is to iteratively solve successively more constrained versions of problem OP 3. That is, a working set of indices $\mathcal{W}\subseteq\{0,1\}^m$ is maintained such that, at each

iteration, the solution $(\mathbf{w}, \xi)$ is only required to satisfy the constraints $\frac{1}{m}\sum_{i=1}^{m} c_i - \langle \mathbf{w}, \Psi_c \rangle \leq \xi$ for $\mathbf{c} \in \mathcal{W}$. The inner **for** loop then finds a new index $\mathbf{c}^*$ which corresponds to the maximally violated constraint in OP 3, and this index is added to the working set. The algorithm proceeds until no constraint is violated by more than ϵ. It can be shown that each iteration must improve the value of the dual objective by a constant amount, from which it follows that the algorithm terminates in a number of rounds independent of m.

Theorem 2 (Joachims [1])**.** *Algorithm 1 terminates after at most* $\max\left\{\frac{2}{\epsilon}, \frac{8CR^2}{\epsilon^2}\right\}$ *iterations, where* $R \stackrel{\text{def}}{=} \max_i \|\mathbf{x}_i\|$ *is the largest* ℓ_2*-norm of the training set vectors. For any training set* S *and any* $\epsilon > 0$*, if* $(\mathbf{w}^*, \xi^*)$ *is an optimal solution of OP 3, then Algorithm 1 returns a point* $(\mathbf{w}, \xi)$ *that has a better objective value than* $(\mathbf{w}^*, \xi)$*, and for which* $(\mathbf{w}, \xi + \epsilon)$ *is feasible in OP 3.*

In terms of time cost, each iteration t of the algorithm involves solving the restricted optimization problem

$$\begin{aligned}(\mathbf{w}, \xi) = \arg\min_{\mathbf{w}, \xi \geq 0} \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle + C\xi \\ \text{s.t.} \quad \forall \mathbf{c} \in \mathcal{W}: \quad \frac{1}{m}\sum_{i=1}^{m} c_i - \langle \mathbf{w}, \Psi_c \rangle \leq \xi\end{aligned}$$

which is done in practice by solving the corresponding dual problem, i.e. the same as OP 4 but with summations over $\mathbf{c} \in \mathcal{W}$ instead of over all $\mathbf{c} \in \{0,1\}$. This involves computing

- $O(t^2)$ matrix elements $J_{\mathbf{c}\mathbf{c}'} = \langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle$
- m inner products $\langle \mathbf{w}, \mathbf{x}_i \rangle = \langle \sum_{\mathbf{c}} \alpha_{\mathbf{c}} \Psi_{\mathbf{c}}, \mathbf{x}_i \rangle$

where $\alpha_{\mathbf{c}}$ is the solution to the dual of the optimization problem in the body of Algorithm 1. In the case of linear SVMs, $\Psi_{\mathbf{c}} = \frac{1}{m}\sum_{i=1}^{m} c_i y_i \mathbf{x}_i$ and $\langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle$ can each be explicitly computed in time $O(m)$. The cost of computing matrix J is $O(mt^2)$, and subsequently solving the dual takes time polynomial in t. As Joachims showed that $t \leq \max\left\{\frac{2}{\epsilon}, \frac{8C\max_i \|\mathbf{x}_i\|^2}{\epsilon^2}\right\}$, and since $\langle \mathbf{w}, \mathbf{x}_i \rangle$ can be computed in time $O(d)$, the entire algorithm therefore has running time linear in m.

For nonlinear SVMs, the feature maps $\Phi(\mathbf{x}_i)$ may be of very large dimension, which precludes explicitly computing $\Psi_{\mathbf{c}} = \frac{1}{m}\sum_{i=1}^{m} c_i y_i \Phi(\mathbf{x}_i)$. Instead, one must compute $J_{\mathbf{c}\mathbf{c}'} = \langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle = \frac{1}{m^2}\sum_{i,j=1}^{m} c_i c_j y_i y_j \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$ as a sum of $O(m^2)$ inner products. This rules out the possibility of an $O(m)$ algorithm, at least using methods that rely on the kernel trick to evaluate each $\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$. Noting that $\mathbf{w} = \sum_{\mathbf{c}} \alpha_{\mathbf{c}} \Psi_{\mathbf{c}}$, the inner products $\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle$ are similarly expensive to compute directly classically if the dimension of the feature map is large.

B. Quantum feature maps

We now show how quantum computing can be used to efficiently approximate the inner products $\langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle$ and $\langle \sum_{\mathbf{c}} \alpha_{\mathbf{c}} \Psi_{\mathbf{c}}, \Phi(\mathbf{x}_i) \rangle$, where high dimensional $\Psi_{\mathbf{c}}$ can be implemented by a quantum circuit using only a number of qubits logarithmic in the dimension. For simplicity, assume binary input vectors, i.e. $\mathbf{x}_i \in \{0,1\}^n$. The generalization to real valued input vectors is straightforward.

Definition 2 (Quantum feature map)**.** *Let $\mathcal{H}_A = \left(\mathbb{C}^2\right)^{\otimes n}, \mathcal{H}_B = \left(\mathbb{C}^2\right)^{\otimes N}$ be n-qubit input and N-qubit output registers respectively . A quantum feature map is a unitary mapping $U_\Phi : \mathcal{H}_A \otimes \mathcal{H}_B \to \mathcal{H}_A \otimes \mathcal{H}_B$ satisfying*

$$U_\Phi |\mathbf{x}\rangle |0\rangle = |\mathbf{x}\rangle |\Phi(\mathbf{x})\rangle ,$$

for each of the basis states $\mathbf{x} \in \{0,1\}^n$, where $|\Phi(\mathbf{x})\rangle = \frac{1}{\|\Phi(\mathbf{x})\|} \sum_{j=1}^{2^N} \Phi(\mathbf{x})_j |j\rangle$, with real amplitudes $\Phi(\mathbf{x})_j \in \mathbb{R}$. Denote the running time of U_Φ by T_Φ.

Note that the states $|\Phi(\mathbf{x})\rangle$ are not necessarily orthogonal. Implementing such a quantum feature map could be done, for instance, through a controlled parameterized quantum circuit.

We also define the quantum state analogy of $\Psi_{\mathbf{c}}$ from Definition 1:

Definition 3. *Given a quantum feature map U_Φ, define $|\Psi_{\mathbf{c}}\rangle$ as*

$$|\Psi_{\mathbf{c}}\rangle = \frac{1}{\|\Psi_{\mathbf{c}}\|} \sum_{i=1}^{m} \frac{c_i y_i}{m} \|\Phi(\mathbf{x}_i)\| \, |\Phi(\mathbf{x}_i)\rangle$$

where

$$\|\Psi_{\mathbf{c}}\|^2 = \frac{1}{m^2} \sum_{i,j=1}^{m} c_i c_j y_i y_j \|\Phi(\mathbf{x}_i)\| \, \|\Phi(\mathbf{x}_j)\| \, \langle \Psi_{\mathbf{c}} | \Psi_{\mathbf{c}'} \rangle,$$

C. Quantum inner product estimation

Let real vectors $x, y \in \mathbb{R}^N$ have corresponding normalized quantum states $|x\rangle = \frac{1}{\|x\|} \sum_{i=1}^{N} x_i |i\rangle$ and $|y\rangle = \frac{1}{\|y\|} \sum_{i=1}^{N} y_i |i\rangle$. The following result shows how the inner product $\langle x, y \rangle = \langle x|y\rangle \|x\| \|y\|$ can be estimated efficiently on a quantum computer.

Theorem 3 (Robust Inner Product Estimation [28], restated)**.** *Let $|x\rangle$ and $|y\rangle$ be quantum states with real amplitudes and with bounded norms $\|x\|, \|y\| \leq R$. If $|x\rangle$ and $|y\rangle$ can each be created in time T, and if estimates of the norms are known to within $\epsilon/3R$ additive error, then one can perform the mapping $|x\rangle |y\rangle |0\rangle \to |x\rangle |y\rangle |s\rangle$ where, with probability at least $1 - \delta$, $|s - \langle x, y \rangle| \leq \epsilon$. The time required to perform this mapping is $\widetilde{O}\left(\frac{R^2 \log(1/\delta)}{\epsilon} T\right)$.*

Thus, if one can efficiently create quantum states $|\Psi_{\mathbf{c}}\rangle$ and estimate the norms $\|\Psi_{\mathbf{c}}\|$, then the corresponding $J_{\mathbf{cc}'} = \langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle = \|\Psi_{\mathbf{c}}\| \|\Psi_{\mathbf{c}'}\| \langle \Psi_{\mathbf{c}} | \Psi_{\mathbf{c}'} \rangle$ can be approximated efficiently. In this section we show that this is possible with a quantum random access memory (qRAM), which is a device that allows for classical data be queried efficiently in superposition. That is, if $x \in \mathbb{R}^N$ is stored in qRAM, then a query to the qRAM implements the unitary $\sum_j \alpha_j |j\rangle |0\rangle \to \sum_j \alpha_j |j\rangle |x_j\rangle$. If the elements x_j of x arrive as a stream of entries (j, x_j) in some arbitrary order, then x can be stored in a particular data structure [29] in time $\tilde{O}(N)$ and, once stored, $|x\rangle = \frac{1}{\|x\|} \sum_j x_j |j\rangle$ can be created in time polylogarithmic in N.

Theorem 4. *Let $\mathbf{c}, \mathbf{c}' \in \{0,1\}^m$. If, for all $i \in [m]$, $\mathbf{x}_i, \frac{c_i y_i}{\sqrt{m}} \|\Phi(\mathbf{x}_i)\|, \frac{c'_i y_i}{\sqrt{m}} \|\Phi(\mathbf{x}_i)\|$ are stored in qRAM, and if $\eta_{\mathbf{c}} = \sqrt{\frac{\sum_{i=1}^m c_i \|\Phi(\mathbf{x}_i)\|^2}{m}}$ and $\eta_{\mathbf{c}'} = \sqrt{\frac{\sum_{i=1}^m c'_i \|\Phi(\mathbf{x}_i)\|^2}{m}}$ are known then, with probability at least $1-\delta$, an estimate $s_{\mathbf{cc}'}$ satisfying*

$$|s_{\mathbf{cc}'} - \langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle| \leq \epsilon$$

can be computed in time

$$T_{cc'} = \tilde{O}\left(\frac{\log(1/\delta)}{\epsilon} \frac{R^3}{\min\{\|\Psi_{\mathbf{c}}\|, \|\Psi_{\mathbf{c}'}\|\}} T_\Phi \right) \tag{1}$$

where $R = \max_i \|\Phi(\mathbf{x}_i)\|$.

A similar result applies to estimating inner products of the form $\sum_{\mathbf{c}\alpha_c} \langle \Psi_{\mathbf{c}}, y_i \Phi(\mathbf{x}_i) \rangle$.

Theorem 5. *Let $\mathcal{W} \subseteq \{0,1\}^m$ and $\sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} \leq C$. If $\eta_{\mathbf{c}}$ are known for all $\mathbf{c} \in \mathcal{W}$ and if $\mathbf{x}_i$ and $\frac{c_i y_i}{\sqrt{m}} \|\Phi(\mathbf{x}_i)\|$ are stored in qRAM for all $i \in [m]$ then, with probability at least $1-\delta$, $\sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} \langle \Psi_{\mathbf{c}}, y_i \Phi(\mathbf{x}_i) \rangle$ can be estimated to within error ϵ in time*

$$\tilde{O}\left(\frac{\log(1/\delta)}{\epsilon} \frac{CR^3 |\mathcal{W}|}{\min_{\mathbf{c}\in\mathcal{W}} \|\Psi_{\mathbf{c}}\|} T_\Phi \right)$$

Proofs of Theorems 4 and 5 are given in Appendix A.

IV. LINEAR TIME ALGORITHM FOR NONLINEAR SVMS

The results of the previous section can be used to generalize Joachims's algorithm to quantum feature mapped data. Let S_n^+ denote the cone of $n \times n$ positive semi-definite matrices. Given $X \in \mathbb{R}^{n\times n}$, let $P_{S_n^+}(X) = \arg\min_{Y\in S_n^+} \|Y - X\|_F$, i.e. the projection of X onto S_n^+, where $\|\cdot\|_F$ is the Frobenius norm. Denote the i-th row of X by $(X)_i$.

Define $IP_{\epsilon,\delta}(x,y)$ to be a quantum subroutine which, with probability at least $1-\delta$, returns an estimate s of the inner product of two vectors x, y satisfying $|s - \langle x, y\rangle| \leq \epsilon$. As we have seen, with appropriate data stored in qRAM, this subroutine can be implemented efficiently on a quantum computer.

Our quantum algorithm for nonlinear structural SVMs is presented in Algorithm 2. At first site, it appears significantly more complicated than Algorithm 1, but this is due in part to more detailed notation used to aid the analysis later. The key differences are (i) the matrix elements $J_{\mathbf{cc}'} = \langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle$ are only estimated to precision ϵ_J by the quantum subroutine; (ii) as the corresponding matrix J is not guaranteed to be positive semi-definite, an additional classical projection step must therefore be carried out to map the estimated matrix on to the p.s.d. cone at each iteration; (iii) In the classical algorithm, the values of c_i^* are deduced by $c_i^* = \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i))$ whereas here we can only estimate the inner products $\left\langle \mathbf{w}^T, \Phi(\mathbf{x}_i) \right\rangle$ to precision ϵ, and $\mathbf{w}$ is known only implicitly according to $\mathbf{w} = \sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} \Psi_{\mathbf{c}}$. Note that apart from the quantum inner product estimation subroutines, all other computations are performed classically.

Theorem 6. *Let $t_{\max}$ be a user-defined parameter and let $(\mathbf{w}^*, \xi^*)$ be an optimal solution of OP 3. If Algorithm 2 terminates in at most $t_{\max}$ iterations then, with probability at least $1-\delta$, it outputs $\hat{\alpha}$ and $\hat{\xi}$ such that $\hat{\mathbf{w}} = \sum_{\mathbf{c}} \hat{\alpha}_{\mathbf{c}} \Psi_{\mathbf{c}}$ satisfies $P(\hat{\mathbf{w}}, \hat{\xi}) - P(\mathbf{w}^*, \xi^*) \leq \min\left\{\frac{C\epsilon}{2}, \frac{\epsilon^2}{8R^2}\right\}$, and $(\hat{\mathbf{w}}, \hat{\xi} + 3\epsilon)$ is feasible for OP 3. If $t_{\max} \geq \max\left\{\frac{4}{\epsilon}, \frac{16CR^2}{\epsilon^2}\right\}$ then the algorithm is guaranteed to terminate in at most $t_{\max}$ iterations.*

Proof. See Appendix B. □

Theorem 7. *Algorithm 2 has a time complexity of*

$$\tilde{O}\left(\frac{CR^3 \log(1/\delta)}{\Psi_{\min}}\left(\frac{t_{\max}^2}{\epsilon} \cdot m + t_{\max}^5\right) T_\Phi\right) \tag{2}$$

where $\Psi_{\min} = \min_{\mathbf{c}\in\mathcal{W}_{t_f}} \|\Psi_{\mathbf{c}}\|$, and $t_f \leq t_{\max}$ is the iteration at which the algorithm terminates.

Proof. See Appendix C.

□

The total number of outer-loop iterations (indexed by t) of Algorithm 2 is upper-bounded by the choice of $t_{\max}$. One may wonder why we do not simply set $t_{\max} = \max\left\{\frac{4}{\epsilon}, \frac{16CR^2}{\epsilon^2}\right\}$ as this would ensure that, with high probability, the algorithm outputs a nearly optimal solution. The reason is that $t_{\max}$ also affects the the quantities $\epsilon_J = \frac{1}{Ctt_{\max}}$, $\delta_J = \frac{\delta}{2t^2 t_{\max}}$ and $\delta_\zeta = \frac{\delta}{2mt_{\max}}$. These in turn

Algorithm 2 Quantum-classical structural SVM algorithm

Input: Training set $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$. SVM hyperparameter C. Quantum feature map U_Φ with maximum norm $R = \max_i \|\Phi(\mathbf{x}_i)\|$. Tolerance parameters $\epsilon, \delta > 0$. $\mathbf{c} \in \{0,1\}^m$. $t_{\max} \geq 1$.

set $t \leftarrow 1$ and $\mathcal{W}_1 \leftarrow \{\mathbf{c}\}$

for $i = 1, \ldots, m$ **do**

Store $\frac{c_i y_i}{\sqrt{m}} \|\Phi(\mathbf{x}_i)\|$ and $\mathbf{x}_i$ in qRAM

Compute and store $\eta_{\mathbf{c}} = \sqrt{\frac{\sum_{i=1}^m c_i \|\Phi(\mathbf{x}_i)\|^2}{m}}$ classically

repeat

set $\epsilon_J \leftarrow \frac{1}{Ct\, t_{\max}}$ and $\delta_J \leftarrow \frac{\delta}{2t^2\, t_{\max}}$

for $\mathbf{c}, \mathbf{c}' \in \mathcal{W}_t$ **do**

$\tilde{J}_{\mathbf{c}\mathbf{c}'} \leftarrow IP_{\epsilon_J, \delta_J}(\Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'})$

$\hat{J}_{\mathcal{W}_t} \leftarrow P_{S^+_{|\mathcal{W}_t|}}(\tilde{J})$

$\hat{\alpha}^{(t)} \leftarrow \text{argmax}_{\alpha \geq 0} -\frac{1}{2} \sum_{\mathbf{c},\mathbf{c}' \in \mathcal{W}} \alpha_{\mathbf{c}} \alpha_{\mathbf{c}'} \left(\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}\mathbf{c}'} + \sum_{\mathbf{c} \in \mathcal{W}} \frac{\|\mathbf{c}\|_1}{m} \alpha_{\mathbf{c}}$

s.t. $\sum_{\mathbf{c} \in \mathcal{W}_t} \alpha_{\mathbf{c}} \leq C$

Store $\hat{\alpha}^{(t)}$ in qRAM

for $\mathbf{c} \in \mathcal{W}_t$ **do**

$\xi_{\mathbf{c}}^{(t)} \leftarrow \max\left\{\frac{1}{m} \sum_{i=1}^m c_i - \sum_{\mathbf{c}' \in \mathcal{W}_t} \hat{\alpha}_{\mathbf{c}'}^{(t)} \left(\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}\mathbf{c}'}, 0\right\}$

set $\hat{\xi}^{(t)} \leftarrow \max_{\mathbf{c} \in \mathcal{W}_t} \xi_{\mathbf{c}}^{(t)} + \frac{1}{t_{\max}}$ and $\delta_\zeta \leftarrow \frac{\delta}{2m\, t_{\max}}$

for $i = 1, \ldots, m$ **do**

$\zeta_i \leftarrow IP_{\epsilon, \delta_\zeta}\left(\sum_{\mathbf{c} \in \mathcal{W}_t} \hat{\alpha}_{\mathbf{c}}^{(t)} \Psi_{\mathbf{c}}, y_i \Phi(\mathbf{x}_i)\right)$

$$c_i^{(t+1)} \leftarrow \begin{cases} 1 & \zeta_i < 1 \\ 0 & \text{otherwise} \end{cases}$$

Store $\frac{c_i^{(t+1)} y_i}{\sqrt{m}} \|\Phi(\mathbf{x}_i)\|$ in qRAM

Compute and store $\eta_{\mathbf{c}^{(t+1)}}$ classically

set $\mathcal{W}_{t+1} \leftarrow \mathcal{W}_t \cup \{\mathbf{c}^{(t+1)}\}$ and $t \leftarrow t + 1$

until $\frac{1}{m} \sum_{i=1}^m \max\{0, 1 - \zeta_i\} \leq \hat{\xi}^{(t)} + 2\epsilon$ OR $t > t_{\max}$

Output: $\hat{\alpha} = \hat{\alpha}^{(t)}, \hat{\xi} = \hat{\xi}^{(t)}$

impact the running time of the two quantum inner product estimation subroutines that take place in each iteration, e.g. the first quantum inner product estimation subroutine has running time that scales like $\frac{\log(1/\delta_J)}{\epsilon_J}$. While the upper-bound on $t_{\max}$ of $\max\left\{\frac{4}{\epsilon}, \frac{16CR^2}{\epsilon^2}\right\}$ is independent of m, it can be large for reasonable values of the other algorithm parameters C, ϵ, δ and R. For instance, the choice of $(C, \epsilon, \delta, R) = (10^4, 0.01, 0.1, 1)$ which, as we show in the Simulation section, lead to

good classification performance on the datasets we consider, corresponds to $t_{\max} = 1.6 \times 10^9$, and $\frac{\log(1/\delta_J)}{\epsilon_J} \geq 1.6 \times 10^{13}$. In practice, we find that this upper-bound on $t_{\max}$ is very loose, and the situation is far better in practice: the algorithm can terminate successfully in very few iterations, with much smaller values of $t_{\max}$. In the examples we consider the algorithm terminates successfully before t reaches $t_{\max} = 50$, corresponding to $\frac{\log(1/\delta_J)}{\epsilon_J} \leq 3.7 \times 10^8$.

The running time of Algorithm 2 also depends on the quantity $\Psi_{\min}$ which is a function of both the dataset as well as the quantum feature map chosen. While this can make $\Psi_{\min}$ hard to predict, we will again see in the Simulation section that in practice the situation is optimistic: we empirically find that $\Psi_{\min}$ is neither too small, nor does it scale noticeably with m or the dimension of the quantum feature map.

A. Classification of new test points

As is standard in SVM theory, the solution $\hat{\alpha}$ from Algorithm 2 can be used to classify a new data point $\mathbf{x}$ according to

$$y_{pred} = \text{sgn} \left\langle \sum_{\mathbf{c}} \hat{\alpha}_{\mathbf{c}} \Psi_{\mathbf{c}}, \Phi(\mathbf{x}) \right\rangle$$

where y_{pred} is the predicted label of $\mathbf{x}$. From Theorem 5, and noting that $|\mathcal{W}| \leq t_{\max}$, we obtain the following result:

Theorem 8. *Let $\hat{\alpha}$ be the output of Algorithm 2, and let $\mathbf{x}$ be stored in qRAM. There is a quantum algorithm that, with probability at least $1-\delta$, estimates the inner product $\langle \sum_{\mathbf{c}} \hat{\alpha}_{\mathbf{c}} \Psi_{\mathbf{c}}, \Phi(\mathbf{x}) \rangle$ to within error ϵ in time $\tilde{O}\left(\frac{CR^3 \log(1/\delta)}{\Psi_{\min}} \frac{t_{\max}}{\epsilon} T_\Phi\right)$*

Taking the sign of the output then completes the classification.

V. SIMULATION

While the true performance of our algorithms for large m and high dimensional quantum feature maps necessitate a fault-tolerant quantum computer to evaluate, we can gain some insight into how it behaves by performing smaller scale numerical experiments on a classical computer. In this section we empirically find that the algorithm can have good performance in practice, both in terms of classification accuracy as well as in terms of the parameters which impact running time.

A. Data set

To test our algorithm we need to choose both a data set as well as a quantum feature map. The general question of what constitutes a good quantum feature map, especially for classifying classical data sets, is an open problem and beyond the scope of this investigation. However, if the data is generated from a quantum problem, then physical intuition may guide our choice of feature map. We therefore consider the following toy example which is nonetheless instructive. Let H_N be the Hamiltonian of a generalized Ising Hamiltonian on N spins

$$H_N(\vec{J},\vec{\Delta},\vec{\Gamma}) = -\sum_{j=1}^{N} J_j Z_j \otimes Z_{j+1} + \sum_{j=1}^{N} (\Delta_j X_j + \Gamma_j Z_j) \tag{3}$$

where $\vec{J},\vec{\Delta},\vec{\Gamma}$ are vectors of real parameters to be chosen, and Z_j, X_j are Pauli Z and X operators acting on the j-th qubit in the chain, respectively. We generate a data set by randomly selecting m points $(\vec{J},\vec{\Delta},\vec{\Gamma})$ and labelling them according to whether the expectation value of the operator $M = \frac{1}{N}\left(\sum_j Z_j\right)^2$ with respect to the ground state of $H_N(\vec{J},\vec{\Delta},\vec{\Gamma})$ satisfies

$$\langle M\rangle \quad \begin{cases} \geq \mu_0 & (+1 \text{ labels}) \\ < \mu_0 & (-1 \text{ labels}) \end{cases} \tag{4}$$

for some cut-off value μ_0, i.e. the points are labelled depending on whether the average total magnetism squared is above or below μ_0. In our simulations we consider a special case of (3) where $J_j = J\cos\frac{k_J\pi(j-1)}{N}$, $\Delta_j = \Delta\sin\frac{k_\Delta\pi j}{N}$ and $\Gamma_j = \Gamma$, where $J, k_J, \Delta, k_\Delta, \Gamma$ are real. Examples of data sets $S_{N,m}$ corresponding to such a Hamiltonian, whose parameters we notate by

$$S_{N,m}(\mu_0, J, k_J, \Delta, k_\Delta, \Gamma),$$

can be found in Fig 1.

B. Quantum feature map

For quantum feature map we choose

$$\left|\Psi(\vec{J},\vec{\Delta},\vec{\Gamma})\right\rangle = \frac{|0\rangle|0\rangle|0\rangle + |1\rangle|\psi_{GS}\rangle|\psi_{GS}\rangle}{\sqrt{2}} \tag{5}$$

where $|\psi_{GS}\rangle$ is the ground state of (3) and, as it is a normalized state, has corresponding value of $R = 1$. We compute such feature maps classically by explicitly diagonalizing H_N. In a real implementation of our algorithm on a quantum computer, such a feature map would be implemented

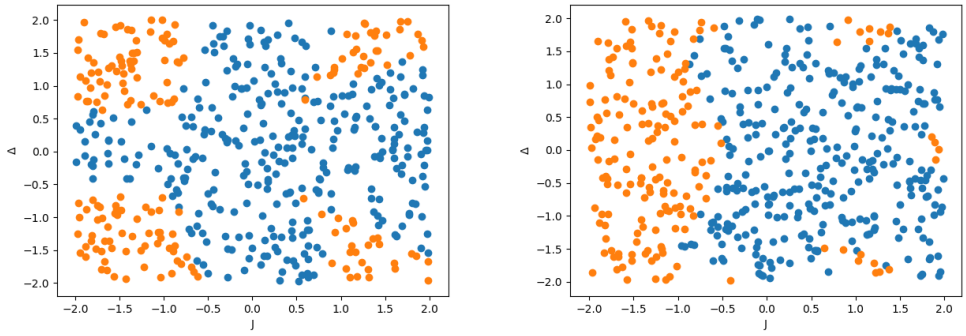

FIG. 1. Sample data sets (left) $S_{5,500}(1.5, J, 1, \Delta, 3, 0.5)$ and (right) $S_{8,500}(2.4, J, 1, \Delta, 2, 0.5)$. Blue and orange colors indicated $+1$ and -1 labels respectively. In each case 500 data points (J, Δ) were generated uniformly at random in the range $[-2, 2]^2$.

by a controlled unitary for generating the (approximate) ground state of H_N, which could be done by a variety of methods e.g. by digitized adiabatic evolution or methods based on imaginary time evolution [30, 31], with running time T_ϕ dependent on the degree of accuracy required. The choice of (5) is motivated by noting that condition (4) is equivalent to determining the sign of $\langle W, \Psi \rangle$, where W is a vector which depends only on μ_0, and not on the choice of parameters in H_N (see Appendix E). By construction, W defines a separating hyperplane for the data, so the chosen quantum feature map separate the data in feature space. As the Hamiltonian is real, it has a set of real eigenvectors and hence $|\Psi\rangle$ can be defined to have real amplitudes, as required.

C. Numerical results

We first evaluate the performance of our algorithm on data sets $S_{N,m}$ for $N = 6$ and increasing orders of m from 10^2 to 10^5.

- For each value of m, a data set $S_{6,m}(\mu_0, J, k_J, \Delta, k_\Delta, \Gamma)$ was generated for points (J, Δ) sampled uniformly at random in the range $[-2, 2]^2$.

- The values of $\mu_0, k_j, k_\Delta, \Gamma$ were fixed and chosen to give roughly balanced data, i.e. the ratio of $+1$ to -1 labels is no more than 70:30 in favor of either label.

- Each set of m data points was divided into training and test sets in the ratio 70:30, and training was performed according to Algorithm 2 with parameters $(C, \epsilon, \delta, t_{\max}) = (10^4, 10^{-2}, 10^{-1}, 50)$.

- These values of C and ϵ were selected to give classification accuracy competitive with classical SVM algorithms utilizing standard Gaussian radial basis function (RBF) kernels, with hyperparameters trained using a subset of the training set of size 20% used for hold-out validation. Note that the quantum feature maps do not have any similar tunable parameters, and a modification of (5), for instance to include a tunable weighting between the two parts of the superposition, could be introduced to further improve performance.

- The quantum $IP_{\epsilon,\delta}$ inner product estimations in the algorithm were approximated by adding Gaussian random noise to the true inner product, such that the resulting inner product was within ϵ of the true value with probability at least $1-\delta$. Classically simulating quantum $IP_{\epsilon,\delta}$ inner product estimation with inner products distributed according to the actual quantum procedures underlying Theorems 4 and 5 was too computationally intensive in general to perform. However, these were tested on small data sets and quantum feature vectors, and found to behave very similarly to adding Gaussian random noise. This is consistent with the results of the numerical simulations in [28].

Note that the values of C, ϵ, δ chosen correspond to $\max\left\{\frac{4}{\epsilon}, \frac{16CR^2}{\epsilon^2}\right\} > 10^9$. This is an upper-bound on the number of iterations $t_{\max}$ needed for the algorithm to converge to a good solution. However, we find empirically that $t_{\max} = 50$ is sufficient for the algorithm to terminate with a good solution across the range of m we consider.

The results are shown in Table I. We find that (i) with these choices of $C, \epsilon, \delta, t_{\max}$ our algorithm has high classification accuracy, competitive with standard classical SVM algorithms utilizing RBF kernels with optimized hyperparameters. (ii) $\Psi_{\min}$ is of the order 10^{-2} in these cases, and does scale noticeably over the range of m from 10^2 to 10^5. If $\Psi_{\min}$ were to decrease polynomially (or worse, exponentially) in m then this would be a severe limitation of our algorithm. Fortunately this does not appear to be the case.

We further investigate the behaviour of $\Psi_{\min}$ by generating data sets $S_{N,m}$ for fixed $m = 1000$ and N ranging from 4 to 8. For each N, we generate 100 random data sets $S_{N,m}(\mu_0, J, k_J, \Delta, k_\Delta, \Gamma)$, where each data set consists of 1000 points (J, Δ) sampled uniformly at random in the range $[-2, 2]^2$, and random values of $\mu_o, k_J, k_\Delta, \Gamma$ chosen to give roughly balanced data sets as before. Unlike before, we do not divide the data into training and test sets. Instead, we perform training on the entire data set, and record the value of $\Psi_{\min}$ in each instance. The results are given in Table II and show that across this range of N (i) the average value $\bar{\Psi}_{\min}$ is of order 10^{-2} (ii) the spread around this average is fairly tight, and the minimum value of $\Psi_{\min}$ in any single instance

m	10^2	10^3	10^4	10^5
$\Psi_{\min}$	0.010	0.018	0.016	0.011
iterations	36	38	39	38
accuracy (%)	93.3	99.3	99.0	98.9
RBF accuracy (%)	86.7	96.0	99.0	99.7

TABLE I. $\Psi_{\min}$, iterations t until termination, and classification accuracy of Algorithm 2 on data sets with parameters $S_{6,m}(1.8, J, 1, \Delta, 9, 0.2)$ for m randomly chosen points $(J, \Delta) \in [-2, 2]^2$, with m in the range $m = 10^2$ to $m = 10^5$. Algorithm parameters were chosen to be $(C, \epsilon, \delta, t_{\max}) = (10^4, 10^{-2}, 10^{-1}, 50)$. The classification accuracy of classical SVMs with Gaussian radial basis function kernels and optimized hyperparameters is given for comparison.

N	4	5	6	7	8
$\bar{\Psi}_{\min}(10^{-2})$	1.28	1.34	1.32	1.44	1.16
$\min(\Psi_{\min})$ (10^{-3})	3.41	2.33	3.16	3.82	3.96
s.d. (10^{-3})	8.6	20.4	16.2	19.6	6.4

TABLE II. Average value, minimum value, and standard deviation of $\Psi_{\min}$ for random data sets generated $S_{N,1000}(\mu_0, J, k_J, \Delta, k_\Delta, \Gamma)$. For each value of N, 100 instances (of $m = 1000$ data points each) were generated for random values of k_J, k_Δ and Γ, with $\mu_0 = 3N$. Algorithm 2 was trained using parameters $(C, \epsilon, \delta, t_{\max}) = (10^4, 10^{-2}, 10^{-1}, 50)$.

is of order 10^{-3}. These support the results of the first experiment, and indicate that the value of $\Psi_{\min}$ may not adversely affect the running time of the algorithm in practice.

VI. CONCLUSIONS

We have proposed a quantum algorithm for training nonlinear soft-margin ℓ_1-SVMs in time linear in the number of training examples m, up to polylogarithmic factors, and given numerical evidence that the algorithm can perform well in practice as well as in theory. This goes beyond previous classical algorithms – which only achieve linear m scaling for linear SVMs or feature maps corresponding to shift-invariant kernels – and previous quantum algorithms – which achieve linear or better scaling in m for other variants of SVMs, which lack some of the desirable properties of the soft-margin ℓ_1-SVM model.

An important direction for future research is to investigate methods for selecting good quantum feature maps for a given problem. While work has been done on learning quantum feature maps by training parameterizable quantum circuits [15, 32–34], a deeper understanding of quantum feature

map construction and optimization is needed. Furthermore, in classical SVM training, typically one of a number of flexible, general purpose kernels such as the Gaussian RBF kernel can be employed in a wide variety of settings. Whether similar, general purpose quantum feature maps can be useful in practice is an open problem, and one that could potentially greatly affect the adoption of quantum algorithms as a useful tool for machine learning.

VII. ACKNOWLEDGEMENTS

We are grateful to Shengyu Zhang for many helpful discussions and feedback on the manuscript.

APPENDIX

Appendix A: Proofs of Theorem 4 and Theorem 5

Lemma 1. *If* $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ *and* $\frac{c_1 y_1}{\sqrt{m}} \|\Phi(\mathbf{x}_1)\|, \ldots, \frac{c_m y_m}{\sqrt{m}} \|\Phi(\mathbf{x}_m)\|$ *for* $\mathbf{c} \in \{0,1\}^m$ *are stored in qRAM, and if* $\eta_{\mathbf{c}} = \sqrt{\frac{\sum_{i=1}^m c_i \|\Phi(\mathbf{x}_i)\|^2}{m}}$ *is known, then* $|\Psi_{\mathbf{c}}\rangle$ *can be created in time* $T_{\Psi_{\mathbf{c}}} = \tilde{O}\left(\frac{R}{\|\Psi_{\mathbf{c}}\|} T_\Phi\right)$*, and* $\|\Psi_{\mathbf{c}}\|$ *estimated to additive error* $\epsilon/3R$ *in time* $O\left(\frac{R^3}{\epsilon} T_\Phi\right)$

Proof. With the above values in qRAM, unitary operators $U_{\mathbf{x}}$ and $U_{\mathbf{c}}$ can be implemented in times $T_{U_{\mathbf{x}}} = \text{polylog}(md)$ and $T_{U_{\mathbf{c}}} = \text{polylog}(m)$, which effect the transformations

$$U_{\mathbf{x}} |i\rangle |0\rangle = |i\rangle |\mathbf{x}_i\rangle$$
$$U_{\mathbf{c}} |0\rangle = \frac{1}{\eta_{\mathbf{c}}} \sum_{j=0}^{m-1} \frac{c_j y_j}{\sqrt{m}} \|\Phi(\mathbf{x}_j)\| |j\rangle$$

$|\Psi_c\rangle$ can then be created by the following procedure:

$$\begin{aligned}
|0\rangle |0\rangle |0\rangle &\xrightarrow{U_{\mathbf{c}}} \frac{1}{\eta_c} \sum_{j=0}^{m-1} \frac{c_j y_j}{\sqrt{m}} \|\Phi(\mathbf{x}_j)\| |j\rangle |0\rangle |0\rangle \\
&\xrightarrow{U_{\mathbf{x}}} \frac{1}{\eta_c} \sum_{j=0}^{m-1} \frac{c_j y_j}{\sqrt{m}} \|\Phi(\mathbf{x}_j)\| |j\rangle |\mathbf{x}_j\rangle |0\rangle \\
&\xrightarrow{U_\Phi} \frac{1}{\eta_c} \sum_{j=0}^{m-1} \frac{c_j y_j}{\sqrt{m}} \|\Phi(\mathbf{x}_j)\| |j\rangle |\mathbf{x}_j\rangle |\Phi(\mathbf{x}_j)\rangle \\
&\xrightarrow{U_{\mathbf{x}}^\dagger} \frac{1}{\eta_c} \sum_{j=0}^{m-1} \frac{c_j y_j}{\sqrt{m}} \|\Phi(\mathbf{x}_j)\| |j\rangle |0\rangle |\Phi(\mathbf{x}_j)\rangle
\end{aligned}$$

Discarding the $|0\rangle$ register, and applying the Hadamard transformation $H |j\rangle = \frac{1}{\sqrt{m}} \sum_k (-1)^{j \cdot k} |k\rangle$ to the first register then gives

$$\begin{aligned}
&\xrightarrow{\text{H}} \frac{1}{\eta_{\mathbf{c}}} \sum_{j=0}^{m-1} \frac{c_j y_j}{\sqrt{m}} \|\Phi(\mathbf{x}_j)\| \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} (-1)^{j \cdot k} |k\rangle |\Phi(\mathbf{x}_j)\rangle \\
&= \frac{\|\Psi_{\mathbf{c}}\|}{\eta_{\mathbf{c}}} |0\rangle \frac{1}{\|\Psi_{\mathbf{c}}\|} \sum_{j=0}^{m-1} \frac{c_j y_j}{m} \|\Phi(\mathbf{x}_j)\| |\Phi(\mathbf{x}_j)\rangle + \left|0^\perp, \text{junk}\right\rangle \\
&= \frac{\|\Psi_{\mathbf{c}}\|}{\eta_{\mathbf{c}}} |0\rangle |\Psi_{\mathbf{c}}\rangle + \left|0^\perp, \text{junk}\right\rangle \qquad \text{(A1)}
\end{aligned}$$

where $\left|0^\perp, \text{junk}\right\rangle$ is an unnormalized quantum state where the first qubit is orthogonal to $|0\rangle$. The state $\frac{\|\Psi_{\mathbf{c}}\|}{\eta_{\mathbf{c}}} |0\rangle |\Psi_{\mathbf{c}}\rangle + \left|0^\perp, \text{junk}\right\rangle$ can therefore be created in time $T_{U_{\mathbf{c}}} + 2T_{U_{\mathbf{x}}} + T_\Phi = \tilde{O}(T_\Phi)$.

By quantum amplitude amplification and amplitude estimation [35], given access to a unitary operator U acting on k qubits such that $U|0\rangle^{\otimes k} = \sin(\theta)|x,0\rangle + \cos(\theta)\left|G,0^{\perp}\right\rangle$ (where $|G\rangle$ is arbitrary), $\sin^2(\theta)$ can be estimated to additive error ϵ in time $O\left(\frac{T(U)}{\epsilon}\right)$ and $|x\rangle$ can be generated in expected time $O\left(\frac{T(U)}{\sin(\theta)}\right)$,where $T(U)$ is the time required to implement U. Amplitude amplification applied to the unitary creating the state in (A1) allows one to create $|\Psi_{\mathbf{c}}\rangle$ in expected time $\tilde{O}\left(\frac{\eta_{\mathbf{c}}}{\|\Psi_{\mathbf{c}}\|}T_{\Phi}\right) = \tilde{O}\left(\frac{R}{\|\Psi_{\mathbf{c}}\|}T_{\Phi}\right)$, since $\eta_c \leq \sqrt{\frac{\sum_{i=1}^{m}\|\Phi(\mathbf{x}_i)^2\|}{m}} \leq R$. Similarly, amplitude estimation can be used to obtain a value s satisfying $\left|s - \frac{\|\Psi_{\mathbf{c}}^2\|}{\eta_{\mathbf{c}}^2}\right| \leq \frac{\epsilon}{3R^3}$ in time $\tilde{O}\left(\frac{R^3}{\epsilon}T_{\Phi}\right)$. Outputting $\overline{\|\Psi_{\mathbf{c}}\|} = \eta_{\mathbf{c}}^2 s$ then satisfies $\left|\overline{\|\Psi_{\mathbf{c}}\|} - \|\Psi_{\mathbf{c}}\|\right| \leq \frac{\epsilon}{3R}$.

□

Theorem 4. *Let $\mathbf{c},\mathbf{c}' \in \{0,1\}^m$. If, for all $i \in [m]$, $\mathbf{x}_i$, $\frac{c_i y_i}{\sqrt{m}}\|\Phi(\mathbf{x}_i)\|$, $\frac{c_i' y_i}{\sqrt{m}}\|\Phi(\mathbf{x}_i)\|$ are stored in qRAM, and if $\eta_{\mathbf{c}} = \sqrt{\frac{\sum_{i=1}^{m} c_i\|\Phi(\mathbf{x}_i)\|^2}{m}}$ and $\eta_{\mathbf{c}'} = \sqrt{\frac{\sum_{i=1}^{m} c_i'\|\Phi(\mathbf{x}_i)\|^2}{m}}$ are known then, with probability at least $1-\delta$, an estimate $s_{\mathbf{c}\mathbf{c}'}$ satisfying*

$$|s_{\mathbf{c}\mathbf{c}'} - \langle\Psi_{\mathbf{c}},\Psi_{\mathbf{c}'}\rangle| \leq \epsilon$$

can be computed in time

$$T_{cc'} = \tilde{O}\left(\frac{\log(1/\delta)}{\epsilon}\frac{R^3}{\min\{\|\Psi_{\mathbf{c}}\|,\|\Psi_{\mathbf{c}'}\|\}}T_{\Phi}\right) \tag{1}$$

where $R = \max_i\|\Phi(\mathbf{x}_i)\|$.

Proof. From Lemma 1, the states $|\Psi_{\mathbf{c}}\rangle$ and $|\Psi_{\mathbf{c}'}\rangle$ can be created in time $\tilde{O}\left(\frac{R}{\min\{\|\Psi_{\mathbf{c}}\|,\|\Psi_{\mathbf{c}'}\|\}}T_{\Phi}\right)$, and estimates of their norms to $\epsilon/3R$ additive error can be obtained in time $\tilde{O}\left(\frac{R^3}{\epsilon}T_{\Phi}\right)$. From Theorem 3 it follows that an estimate $s_{cc'}$ satisfying

$$|s_{\mathbf{c}\mathbf{c}'} - \langle\Psi_{\mathbf{c}},\Psi_{\mathbf{c}'}\rangle| \leq \epsilon$$

can be found with probability at least $1-\delta$ in time

$$T_{est} = \tilde{O}\left(\frac{\log(1/\delta)}{\epsilon}\frac{R^3}{\min\{\|\Psi_{\mathbf{c}}\|,\|\Psi_{\mathbf{c}'}\|\}}T_{\Phi}\right)$$

□

Theorem 5. *Let $\mathcal{W} \subseteq \{0,1\}^m$ and $\sum_{\mathbf{c}\in\mathcal{W}}\alpha_{\mathbf{c}} \leq C$. If $\eta_{\mathbf{c}}$ are known for all $\mathbf{c} \in \mathcal{W}$ and if $\mathbf{x}_i$ and $\frac{c_i y_i}{\sqrt{m}}\|\Phi(\mathbf{x}_i)\|$ are stored in qRAM for all $i \in [m]$ then, with probability at least $1-\delta$, $\sum_{\mathbf{c}\in\mathcal{W}}\alpha_{\mathbf{c}}\langle\Psi_{\mathbf{c}}, y_i\Phi(\mathbf{x}_i)\rangle$ can be estimated to within error ϵ in time*

$$\tilde{O}\left(\frac{\log(1/\delta)}{\epsilon}\frac{CR^3|\mathcal{W}|}{\min_{\mathbf{c}\in\mathcal{W}}\|\Psi_{\mathbf{c}}\|}T_{\Phi}\right)$$

Proof. With the above data in qRAM, an almost identical analysis to that in Theorem 4 can be applied to deduce that, for any $\mathbf{c} \in \mathcal{W}$, with probability at least $1-\delta/|\mathcal{W}|$, an estimate $t_{\mathbf{c}i}$ satisfying

$$|t_{\mathbf{c}i} - \langle \Psi_{\mathbf{c}}, y_i \Phi(\mathbf{x}_i) \rangle| \leq \epsilon/C$$

can be computed in time

$$T_{\mathbf{c}i} = \tilde{O}\left(\frac{C \log(|\mathcal{W}|/\delta)}{\epsilon} \frac{R^3}{\min_{\mathbf{c}\in\mathcal{W}} \|\Psi_{\mathbf{c}}\|} T_\Phi \right)$$

and the total time required to estimate all $|\mathcal{W}|$ terms (i.e. $t_{\mathbf{c}i}$ for all $\mathbf{c} \in \mathcal{W}$) is thus $|\mathcal{W}| T_{\mathbf{c}i}$. The probability that every term $t_{\mathbf{c}i}$ is obtained to ϵ/C accuracy is therefore $(1-\delta/|\mathcal{W}|)^{|\mathcal{W}|} \geq 1-\delta$. In this case, the weighted sum $\sum_{\mathbf{c}\mathcal{W}} \alpha_{\mathbf{c}} t_{\mathbf{c}i}$ can be computed classically, and satisfies

$$\begin{aligned}
\sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} t_{\mathbf{c}i} &\leq \sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} \left(\langle \Psi_{\mathbf{c}}, y_i \Phi(\mathbf{x}_i) \rangle + \epsilon/C \right) \\
&= \sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} \langle \Psi_{\mathbf{c}}, y_i \Phi(\mathbf{x}_i) \rangle + \frac{\epsilon}{C} \sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} \\
&= \sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} \langle \Psi_{\mathbf{c}}, y_i \Phi(\mathbf{x}_i) \rangle + \epsilon
\end{aligned}$$

and similarly $\sum_{\mathbf{c}} \alpha_{\mathbf{c}} t_{\mathbf{c}i} \geq \sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} \langle \Psi_{\mathbf{c}}, y_i \Phi(\mathbf{x}_i) \rangle - \epsilon$. □

Appendix B: Proof of Theorem 6

The analysis of Algorithm 2 is based on [9, 20], with additional steps and complexity required to bound the errors due to inner product estimation and projection onto the p.s.d. cone.

Theorem 6. *Let $t_{\max}$ be a user-defined parameter and let $(\mathbf{w}^*, \xi^*)$ be an optimal solution of OP 3. If Algorithm 2 terminates in at most $t_{\max}$ iterations then, with probability at least $1-\delta$, it outputs $\hat{\alpha}$ and $\hat{\xi}$ such that $\hat{\mathbf{w}} = \sum_{\mathbf{c}} \hat{\alpha}_{\mathbf{c}} \Psi_{\mathbf{c}}$ satisfies $P(\hat{\mathbf{w}}, \hat{\xi}) - P(\mathbf{w}^*, \xi^*) \leq \min\left\{ \frac{C\epsilon}{2}, \frac{\epsilon^2}{8R^2} \right\}$, and $(\hat{\mathbf{w}}, \hat{\xi} + 3\epsilon)$ is feasible for OP 3. If $t_{\max} \geq \max\left\{ \frac{4}{\epsilon}, \frac{16CR^2}{\epsilon^2} \right\}$ then the algorithm is guaranteed to terminate in at most $t_{\max}$ iterations.*

Lemma 2. *When Algorithm 2 terminates successfully after at most $t_{\max}$ iterations, the probability that all inner products are estimated to within their required tolerances throughout the duration of the algorithm is at least $1-\delta$.*

Proof. Each iteration t of the Algorithm involves

- $O(t^2)$ inner product estimations $IP_{\epsilon_J,\delta_J}(\Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'})$, for all pairs $\mathbf{c}, \mathbf{c}' \in \mathcal{W}_t$. The probability of successfully computing all t^2 inner products to within error ϵ_J is at least $(1-\delta_J)^{t^2} = \left(1 - \frac{\delta}{2t^2\, t_{\max}}\right)^{t^2} \geq 1 - \frac{\delta}{2\, t_{\max}}$.
- m inner product estimations $IP_{\epsilon,\delta_\zeta}\left(\sum_{\mathbf{c}\in\mathcal{W}^t} \alpha_{\mathbf{c}}^{(t)} \Psi_{\mathbf{c}}, y_i \Phi(\mathbf{x}_i)\right)$, for $i = 1, \ldots, m$. The probability of all estimates lying within error ϵ is at least $\left(1 - \frac{\delta}{2mt_{\max}}\right)^m \geq 1 - \frac{\delta}{2t_{\max}}$.

Since the algorithm terminates successfully after at most $t_{\max}$ iterations, the probability that all the inner products are estimated to within their required tolerances is

$$\prod_{t=1}^{t_{\max}} \left(1 - \frac{\delta}{2t_{\max}}\right)^2 \geq 1 - \delta$$

where the right hand side follows from Bernoulli's inequality.

□

By Lemma 2 we can analyze Algorithm 2, assuming that all the quantum inner product estimations succeed, i.e. each call to $IP_{\epsilon,\delta}(\mathbf{x}, \mathbf{y})$ produces an estimate of $\langle \mathbf{x}, \mathbf{y} \rangle$ within error ϵ. In what follows, let $J_{\mathcal{W}_t}$ be the $|\mathcal{W}_t| \times |\mathcal{W}_t|$ matrix with elements $\langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle$ for $\mathbf{c}, \mathbf{c}' \in \mathcal{W}$, let $\delta\hat{J}_{\mathcal{W}_t} \stackrel{\mathsf{def}}{=} J_{\mathcal{W}_t} - \hat{J}_{\mathcal{W}_t}$.

Lemma 3. $\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_\sigma \leq \left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_F \leq \frac{1}{Ct_{\max}}$, *where* $\|\cdot\|_\sigma$ *is the spectral norm.*

Proof. The relation between the spectral and Frobenius norms is elementary. We thus prove the upper-bound on the Frobenius norm. By assumption, all matrix elements $\tilde{J}_{\mathbf{c}\mathbf{c}'}$ satisfy $\left|\tilde{J}_{\mathbf{c}\mathbf{c}'} - \langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle\right| \leq \epsilon_J = \frac{1}{Ctt_{\max}}$. Thus,

$$\begin{aligned}
\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_F &= \left\|J_{\mathcal{W}_t} - \hat{J}_{\mathcal{W}_t}\right\|_F \\
&= \left\|J_{\mathcal{W}_t} - P_{S^+_{|\mathcal{W}_t|}}(\tilde{J})\right\|_F \\
&\leq \left\|J_{\mathcal{W}_t} - \tilde{J}_{\mathcal{W}_t}\right\|_F \\
&\leq |\mathcal{W}_t|\, \epsilon_J \\
&\leq \epsilon_J t \\
&= \frac{1}{Ct_{\max}}
\end{aligned}$$

where the second equality follows from the definition of $\hat{J}_{\mathcal{W}_t}$ in Algorithm 2, the first inequality because projecting $\tilde{J}_{\mathcal{W}}$ onto the p.s.d cone cannot increase its Frobenius norm distance to a p.s.d matrix $J_{\mathcal{W}_t}$, and the third inequality because the size of the index set $\mathcal{W}_t$ increases by at most one per iteration. □

To proceed, let us introduce some additional notation. Given index set $\mathcal{W}$, define

$$D_{\mathcal{W}}(\alpha) = -\frac{1}{2}\sum_{\mathbf{c},\mathbf{c}'\in\mathcal{W}} \alpha_{\mathbf{c}}\alpha_{\mathbf{c}'}J_{\mathbf{c}\mathbf{c}'} + \sum_{\mathbf{c}\in\mathcal{W}} \frac{\|\mathbf{c}\|_1}{m}\alpha_{\mathbf{c}}$$
$$\hat{D}_{\mathcal{W}}(\alpha) = -\frac{1}{2}\sum_{\mathbf{c},\mathbf{c}'\in\mathcal{W}} \alpha_{\mathbf{c}}\alpha_{\mathbf{c}'}\hat{J}_{\mathbf{c}\mathbf{c}'} + \sum_{\mathbf{c}\in\mathcal{W}} \frac{\|\mathbf{c}\|_1}{m}\alpha_{\mathbf{c}}$$

and let $D^*_{\mathcal{W}}$ and $\hat{D}^*_{\mathcal{W}}$ be the maximum values of $D_{\mathcal{W}}(\alpha)$ and $\hat{D}_{\mathcal{W}}(\alpha)$ respectively, subject to the constraints $\alpha \geq 0, \sum_{\mathbf{c}\in\mathcal{W}} \alpha_{\mathbf{c}} \leq C$. Since $\hat{J}$ above is positive semi-definite, its matrix elements can expressed as

$$\hat{J}_{\mathbf{c}\mathbf{c}'} = \left\langle \hat{\Psi}_{\mathbf{c}}, \hat{\Psi}_{\mathbf{c}'} \right\rangle \tag{B1}$$

for some set of vectors $\{\hat{\Psi}_{\mathbf{c}}\}$.

The next lemma shows that the solution $\hat{\alpha}^{(t)}$ obtained at each step is only slightly suboptimal as a solution for the restricted problem $D_{\mathcal{W}_t}$.

Lemma 4. $D^*_{\mathcal{W}_t} - \frac{C}{t_{\max}} \leq D_{\mathcal{W}_t}(\hat{\alpha}^{(t)}) \leq D^*_{\mathcal{W}_t}$.

Proof.

$$\begin{aligned}
D^*_{\mathcal{W}_t} &\geq D_{\mathcal{W}_t}\left(\hat{\alpha}^{(t)}\right) \\
&= -\frac{1}{2}\left(\hat{\alpha}^{(t)}\right)^T \left(\hat{J}_{\mathcal{W}_t} + \delta\hat{J}_{\mathcal{W}_t}\right)\hat{\alpha}^{(t)} + \sum_{\mathbf{c}\in\mathcal{W}} \frac{\|\mathbf{c}\|_1}{m}\hat{\alpha}^{(t)}_{\mathbf{c}} \\
&= \hat{D}^*_{\mathcal{W}_t} - \frac{1}{2}\left(\hat{\alpha}^{(t)}\right)^T \left(\delta\hat{J}_{\mathcal{W}_t}\right)\hat{\alpha}^{(t)} \\
&\geq \hat{D}^*_{\mathcal{W}_t} - \frac{1}{2}\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_\sigma \left\|\hat{\alpha}^{(t)}\right\|_2^2 \\
&\geq \hat{D}^*_{\mathcal{W}_t} - \frac{C^2}{2}\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_\sigma
\end{aligned} \tag{B2}$$

The first inequality follows from the fact that $\hat{\alpha}^{(t)}$ is, by definition, optimal for $\hat{D}_{\mathcal{W}_t}$ and feasible for $D_{\mathcal{W}_t}$, and the last inequality comes from the fact that $\left\|\hat{\alpha}^{(t)}\right\|_2 \leq \left\|\hat{\alpha}^{(t)}\right\|_1 \leq C$. Similarly,

$$\hat{D}^*_{\mathcal{W}_t} \geq D^*_{\mathcal{W}_t} - \frac{C^2}{2}\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_\sigma \tag{B3}$$

and the result follows from substituting (B3) into (B2), and using lemma 3. □

We now show that $\hat{\alpha}^{(t)}$ and $\hat{\xi}^{(t)}$ can be used to define a feasible solution for OP3 where the constraints are restricted to only hold over the index set $\mathcal{W}_t$.

Lemma 5. *Define* $\mathbf{w}^{(t)} \stackrel{\text{def}}{=} \sum_{\mathbf{c}\in\mathcal{W}_t} \hat{\alpha}^{(t)}_{\mathbf{c}}\Psi_{\mathbf{c}}$. *It holds that* $\frac{1}{m}\sum_{i=1}^m c_i - \langle \mathbf{w}, \Psi_{\mathbf{c}}\rangle \leq \hat{\xi}^{(t)}$ *for all* $\mathbf{c} \in \mathcal{W}_t$.

Proof. First note that

$$\begin{aligned}\sum_{\mathbf{c}'\in\mathcal{W}_t}\hat{\alpha}^{(t)}_{\mathbf{c}'}\left(\delta\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}^*\mathbf{c}'} &= \left\langle\left(\delta\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}*},\hat{\alpha}^{(t)}\right\rangle\\ &\geq -\left\|\left(\delta\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}*}\right\|_2\left\|\hat{\alpha}^{(t)}\right\|_2\\ &\geq -C\left\|\left(\delta\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}*}\right\|_2\\ &\geq -C\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_F\\ &\geq -\frac{1}{t_{\max}}\end{aligned} \tag{B4}$$

where the second inequality is due to $\left\|\hat{\alpha}^{(t)}\right\|_2 \leq \left\|\hat{\alpha}^{(t)}\right\|_1 \leq C$, the third is because $\left\|\left(\delta\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}*}\right\|_2 \leq \left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_F$, the fourth follows from Lemma 3.

Let $\mathbf{c}^* = \arg\max_{\mathbf{c}\in\mathcal{W}_t}\left(\frac{1}{m}\sum_{i=1}^m c_i - \sum_{\mathbf{c}'\in\mathcal{W}}\hat{\alpha}^{(t)}_{\mathbf{c}'}J_{\mathbf{c}\mathbf{c}'}\right)$. Then,

$$\begin{aligned}\hat{\xi}^{(t)} &\stackrel{\mathsf{def}}{=} \max_{\mathbf{c}\in\mathcal{W}_t}\left(\frac{1}{m}\sum_{i=1}^m c_i - \sum_{\mathbf{c}'\in\mathcal{W}}\hat{\alpha}^{(t)}_{\mathbf{c}'}\left(\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}\mathbf{c}'}\right) + \frac{1}{t_{\max}}\\ &\geq \frac{1}{m}\sum_{i=1}^m c^*_i - \sum_{\mathbf{c}'\in\mathcal{W}_t}\hat{\alpha}^{(t)}_{\mathbf{c}'}\left(\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}^*\mathbf{c}'} + \frac{1}{t_{\max}}\\ &= \frac{1}{m}\sum_{i=1}^m c^*_i - \sum_{\mathbf{c}'\in\mathcal{W}_t}\hat{\alpha}^{(t)}_{\mathbf{c}'}\left(J_{\mathcal{W}_t} - \delta\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}^*\mathbf{c}'} + \frac{1}{t_{\max}}\\ &= \max_{\mathbf{c}\in\mathcal{W}_t}\left(\frac{1}{m}\sum_{i=1}^m c_i - \sum_{\mathbf{c}'\in\mathcal{W}_t}\hat{\alpha}^{(t)}_{\mathbf{c}'}J_{\mathbf{c}\mathbf{c}'}\right) + \sum_{\mathbf{c}'\in\mathcal{W}_t}\hat{\alpha}^{(t)}_{\mathbf{c}'}\left(\delta\hat{J}_{\mathcal{W}_t}\right)_{\mathbf{c}^*\mathbf{c}'} + \frac{1}{t_{\max}}\\ &\geq \max_{\mathbf{c}\in\mathcal{W}_t}\left(\frac{1}{m}\sum_{i=1}^m c_i - \sum_{\mathbf{c}'\in\mathcal{W}_t}\hat{\alpha}^{(t)}_{\mathbf{c}'}J_{\mathbf{c}\mathbf{c}'}\right)\\ &= \max_{\mathbf{c}\in\mathcal{W}_t}\left(\frac{1}{m}\sum_{i=1}^m c_i - \langle\mathbf{w},\Psi_{\mathbf{c}}\rangle\right)\end{aligned}$$

where the last inequality follows from (B4). □

The next lemma shows that at each step which does not terminate the algorithm, the solution $(\hat{\mathbf{w}}^{(t)} \stackrel{\mathsf{def}}{=} \sum_{\mathbf{c}}\hat{\alpha}^{(t)}_{\mathbf{c}}\Psi_{\mathbf{c}}, \hat{\xi}^{(t)})$ violates the constraint indexed by $\mathbf{c}^{(t+1)}$ in OP 3 by at least ϵ.

Lemma 6.

$$\frac{1}{m}\sum_{i=1}^m \max\{0, 1-\zeta_i\} > \hat{\xi}^{(t)} + 2\epsilon \Rightarrow \frac{1}{m}\sum_{i=1}^n c^{(t+1)}_i - \left\langle\mathbf{w}^{(t)}, \Psi_{\mathbf{c}^{(t+1)}}\right\rangle > \hat{\xi}^{(t)} + \epsilon,$$

where $\hat{\mathbf{w}}^{(t)} \stackrel{\mathsf{def}}{=} \sum_{\mathbf{c}}\hat{\alpha}^{(t)}_{\mathbf{c}}\Psi_{\mathbf{c}}$.

Proof. Algorithm (2) assigns the values

$$\zeta_i \leftarrow IP_{\epsilon,\delta_\zeta}\left(\sum_{\mathbf{c}\in\mathcal{W}^t}\hat{\alpha}_{\mathbf{c}}^{(t)}\Psi_{\mathbf{c}}, y_i\Phi(\mathbf{x}_i)\right)$$

$$c_i^{(t+1)} \leftarrow \begin{cases} 1 & \zeta_i < 1 \\ 0 & \text{otherwise} \end{cases}$$

Assuming $\hat{\xi}^{(t)} + 2\epsilon < \frac{1}{m}\sum_{i=1}^m \max\{0, 1-\zeta_i\}$, it follows that

$$\begin{aligned}
\hat{\xi}^{(t)} + 2\epsilon &< \frac{1}{m}\sum_{i=1}^m \max\{0, 1-\zeta_i\} \\
&= \frac{1}{m}\sum_{i=1}^m c_i^{(t+1)}(1-\zeta_i) \\
&= \frac{1}{m}\sum_{i=1}^m c_i^{(t+1)} - \frac{1}{m}\sum_{i=1}^m c_i^{(t+1)} IP_{\epsilon,\delta_\zeta}\left(\sum_{\mathbf{c}\in\mathcal{W}^t}\hat{\alpha}_{\mathbf{c}}^{(t)}\Psi_{\mathbf{c}}, y_i\Phi(\mathbf{x}_i)\right) \\
&\leq \frac{1}{m}\sum_{i=1}^m c_i^{(t+1)} - \frac{1}{m}\sum_{i=1}^m c_i^{(t+1)}\left(\left\langle\sum_{\mathbf{c}\in\mathcal{W}^t}\hat{\alpha}_{\mathbf{c}}^{(t)}\Psi_{\mathbf{c}}, y_i\Phi(\mathbf{x}_i)\right\rangle - \epsilon\right) \\
&= \frac{1}{m}\sum_{i=1}^m c_i^{(t+1)} - \left\langle\sum_{\mathbf{c}\in\mathcal{W}_t}\hat{\alpha}_{\mathbf{c}}^{(t)}\Psi_{\mathbf{c}}, \frac{1}{m}\sum_{i=1}^m c_i^{(t+1)} y_i\Phi(\mathbf{x}_i)\right\rangle + \epsilon \\
&= \frac{1}{m}\sum_{i=1}^m c_i^{(t+1)} - \left\langle\mathbf{w}^{(t)}, \Psi_{\mathbf{c}^{(t+1)}}\right\rangle + \epsilon
\end{aligned}$$

□

Next we show that each iteration of the algorithm increases the working set $\mathcal{W}$ such that the optimal solution of the restricted problem $D_{\mathcal{W}}$ increases by a certain amount. Note that we do not explicitly compute $D^*_{\mathcal{W}}$, as it will be sufficient to know that its value increases each iteration.

Lemma 7. *While* $\xi^{(t)} > \hat{\xi} + \epsilon + \epsilon_c$, $D^*_{\mathcal{W}_{t+1}} - D^*_{\mathcal{W}_t} \geq \min\left\{\frac{C\epsilon}{2}, \frac{\epsilon^2}{8R^2}\right\} - \frac{C}{t_{\max}}$

Proof. Given $\hat{\alpha}^{(t)}$ at iteration t, define $\alpha, \eta \in \mathbb{R}^{2^m}$ by

$$\alpha_{\mathbf{c}} = \begin{cases} \hat{\alpha}_{\mathbf{c}} & \mathbf{c}\in\mathcal{W}_t \\ 0 & \text{o.w.} \end{cases} \qquad \eta_{\mathbf{c}} = \begin{cases} 1 & \mathbf{c} = \mathbf{c}^{(t+1)} \\ -\frac{\alpha_c}{C} & \mathbf{c}\in\mathcal{W}_t \\ 0 & \text{o.w.} \end{cases}$$

For any $0 \leq \beta \leq C$, the vector $\alpha + \beta\eta$ is entrywise non-negative by construction, and satisfies

$$\begin{aligned}\sum_{\mathbf{c}\in\{0,1\}^m} (\alpha + \beta\eta)_{\mathbf{c}} &= \beta + \left(1 - \frac{\beta}{C}\right) \sum_{\mathbf{c}\in\mathcal{W}_t} \hat{\alpha}_{\mathbf{c}} \\ &\leq \beta + C\left(1 - \frac{\beta}{C}\right) \\ &= C\end{aligned}$$

$\alpha + \beta\eta$ is therefore a feasible solution of OP 4. Furthermore, by considering the Taylor expansion of the OP 4 objective function $D(\alpha)$ it is straightforward to show that

$$\max_{0\leq\beta\leq C} (D(\alpha + \beta\eta) - D(\alpha)) \geq \frac{1}{2} \min\left\{C, \frac{\eta^T \nabla D(\alpha)}{\eta^T J \eta}\right\} \eta^T \nabla D(\alpha) \tag{B5}$$

for any η satisfying $\eta^T \nabla D(\alpha) > 0$ (See Appendix D). We now show that this condition holds for the η defined above. The gradient of D satisfies

$$\begin{aligned}\nabla D(\alpha)_{\mathbf{c}} &= \frac{1}{m}\sum_{i=1}^{m} c_i - \sum_{\mathbf{c}'\in\mathcal{W}_t} \alpha_{\mathbf{c}'} J_{\mathbf{c}\mathbf{c}'} \\ &= \frac{1}{m}\sum_{i=1}^{m} c_i - \left\langle \mathbf{w}^{(t)}, \Psi_{\mathbf{c}} \right\rangle\end{aligned}$$

From Lemmas 5 and 6 we have

$$\nabla D(\alpha)_{\mathbf{c}} \begin{cases} \leq \hat{\xi}^{(t)} & \mathbf{c} \in \mathcal{W}_t \\ > \hat{\xi}^{(t)} + \epsilon & \mathbf{c} = \mathbf{c}^{(t+1)} \end{cases}$$

and since $\sum_{\mathbf{c}\in\mathcal{W}_t} \alpha_{\mathbf{c}} = \sum_{\mathbf{c}\in\mathcal{W}_t} \hat{\alpha}_{\mathbf{c}} \leq C$ it follows that

$$\begin{aligned}\eta^T \nabla D(\alpha) &= \left(\hat{\xi}^{(t)} + \epsilon\right) - \frac{1}{C} \sum_{\mathbf{c}\in\mathcal{W}_t} \alpha_{\mathbf{c}} \nabla D(\alpha)_{\mathbf{c}} \\ &\geq \left(\hat{\xi}^{(t)} + \epsilon\right) - \frac{\hat{\xi}^{(t)}}{C} \sum_{c\in\mathcal{W}_t} \alpha_{\mathbf{c}} \\ &= \epsilon\end{aligned} \tag{B6}$$

Also:

$$\begin{aligned}\eta^T J \eta &= J_{\mathbf{c}^{(t+1)}\mathbf{c}^{(t+1)}} + \frac{1}{C^2} \sum_{\mathbf{c},\mathbf{c}'\in\mathcal{W}_t} \alpha_{\mathbf{c}} \alpha_{\mathbf{c}'} J_{\mathbf{c}\mathbf{c}'} - \frac{2}{C} \sum_{\mathbf{c}\in\mathcal{W}_t} \alpha_{\mathbf{c}} J_{\mathbf{c}\mathbf{c}^{(t+1)}} \\ &\leq R^2 + \frac{R}{C^2} \sum_{\mathbf{c},\mathbf{c}'\in\mathcal{W}_t} \alpha_{\mathbf{c}} \alpha_{\mathbf{c}'} + \frac{2R^2}{C} \sum_{\mathbf{c}\in\mathcal{W}_t} \alpha_{\mathbf{c}} \\ &\leq 4R^2\end{aligned} \tag{B7}$$

where we note that $J_{\mathbf{cc}'} = \langle \Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'} \rangle \leq \max_{\mathbf{c} \in \{0,1\}^m} \|\Psi_{\mathbf{c}}\|^2 \leq R^2$. Combining (B5), (B6), (B7) gives

$$\max_{\beta \in [0,C]} D(\alpha + \beta\eta) - D(\alpha) \geq \min \left\{ \frac{C\epsilon}{2}, \frac{\epsilon^2}{8R^2} \right\}$$

By construction $\max_{\beta \in [0,C]} D(\alpha + \beta\eta) \leq D^*_{\mathcal{W}_{t+1}}$ and Lemma 4 gives $D^*_{\mathcal{W}_t} - \frac{C}{t_{\max}} \leq D_{\mathcal{W}_t}(\hat{\alpha})$. Thus

$$\begin{aligned} D^*_{\mathcal{W}_{t+1}} &\geq D(\alpha) + \min \left\{ \frac{C\epsilon}{2}, \frac{\epsilon^2}{8R^2} \right\} \\ &= D_{\mathcal{W}_t}(\hat{\alpha}) + \min \left\{ \frac{C\epsilon}{2}, \frac{\epsilon^2}{8R^2} \right\} \\ &\geq D^*_{\mathcal{W}_t} + \min \left\{ \frac{C\epsilon}{2}, \frac{\epsilon^2}{8R^2} \right\} - \frac{C}{t_{\max}} \end{aligned}$$

□

Corollary 1. *If* $t_{\max} \geq \min \left\{ \frac{4}{\epsilon}, \frac{16CR^2}{\epsilon^2} \right\}$*, Algorithm 2 terminates after at most* $t_{\max}$ *iterations.*

Proof. Lemma 7 shows that the optimal dual objective value $D^*_{\mathcal{W}_t}$ increases by at least $\min \left\{ \frac{C\epsilon}{2}, \frac{\epsilon^2}{8R^2} \right\} - \frac{C}{t_{\max}}$ each iteration. For $t_{\max} \geq \min \left\{ \frac{4}{\epsilon}, \frac{16CR^2}{\epsilon^2} \right\}$, this increase is at least $\frac{C}{t_{\max}}$. $D^*_{\mathcal{W}_t}$ is upperbounded by D^*, the optimal value of OP 4 which, by Lagrange duality, is equal to the optimum value of the primal problem OP 3, which is itself upper bounded by C (corresponding to feasible solution $\mathbf{w} = 0, \xi = 1$). Thus, the algorithm must terminate after at most $t_{\max}$ iterations. □

We know show that the outputs $\hat{\alpha}$ and $\hat{\xi}$ of Algorithm 2 can be used to define a feasible solution to OP 3.

Lemma 8. *Let* $(\hat{\alpha}), \hat{\xi}$ *be the outputs of Algorithm 2 , in the event that the algorithm terminates within* $t_{\max}$ *iterations. Let* $\hat{\mathbf{w}} = \sum_{\mathbf{c}} \hat{\alpha}_{\mathbf{c}} \Psi_{\mathbf{c}}$*. Then* $(\hat{\mathbf{w}}, \hat{\xi} + 3\epsilon)$ *is feasible for OP 3.*

Proof. By construction $\hat{\xi} + 3\epsilon > 0$. The termination condition $\frac{1}{m} \sum_{i=1}^m \max\{0, 1 - \xi_i\} \leq \hat{\xi} + 2\epsilon$

implies that

$$\begin{aligned}
\max_{\mathbf{c}\in\{0,1\}^m}\left(\frac{1}{m}\sum_{i=1}^m c_i - \langle\hat{\mathbf{w}},\Psi_{\mathbf{c}}\rangle\right) &= \max_{\mathbf{c}\in\{0,1\}^m}\left(\frac{1}{m}\sum_{i=1}^m c_i - \frac{1}{m}\sum_{i=1}^m c_i y_i\left\langle\hat{\mathbf{w}},\Phi(\mathbf{x}_i)\right\rangle\right)\\
&= \frac{1}{m}\sum_{i=1}^m \max_{c_i\in\{0,1\}}\left(c_i - c_i\left\langle\hat{\mathbf{w}}, y_i\Phi(\mathbf{x}_i)\right\rangle\right)\\
&\leq \frac{1}{m}\sum_{i=1}^m \max_{c_i\in\{0,1\}} c_i\left(1-\xi_i+\epsilon\right)\\
&\leq \frac{1}{m}\sum_{i=1}^m \max_{c_i\in\{0,1\}} c_i\left(1-\xi_i\right)+\epsilon\\
&= \frac{1}{m}\sum_{i=1}^m \max\left\{0,1-\xi_i\right\}+\epsilon\\
&\leq \hat{\xi}+3\epsilon
\end{aligned}$$

$(\hat{\mathbf{w}},\hat{\xi}+3\epsilon)$ therefore satisfy all the constraints of OP 3. □

We are now in a position to prove Theorem 6.

Theorem 6. *Let $t_{\max}$ be a user-defined parameter and let $(\mathbf{w}^*,\xi^*)$ be an optimal solution of OP 3. If Algorithm 2 terminates in at most $t_{\max}$ iterations then, with probability at least $1-\delta$, it outputs $\hat{\alpha}$ and $\hat{\xi}$ such that $\hat{\mathbf{w}}=\sum_{\mathbf{c}}\hat{\alpha}_{\mathbf{c}}\Psi_{\mathbf{c}}$ satisfies $P(\hat{\mathbf{w}},\hat{\xi})-P(\mathbf{w}^*,\xi^*)\leq\min\left\{\frac{C\epsilon}{2},\frac{\epsilon^2}{8R^2}\right\}$, and $(\hat{\mathbf{w}},\hat{\xi}+3\epsilon)$ is feasible for OP 3. If $t_{\max}\geq\max\left\{\frac{4}{\epsilon},\frac{16CR^2}{\epsilon^2}\right\}$ then the algorithm is guaranteed to terminate in at most $t_{\max}$ iterations.*

Proof. The guarantee of termination with $t_{\max}$ iterations for $t_{\max}\geq\max\left\{\frac{4}{\epsilon},\frac{16CR^2}{\epsilon^2}\right\}$ is given by Corollary 1, and the feasibility of $(\hat{\mathbf{w}},\hat{\xi}+3\epsilon)$ is given by Lemma 8.

Let the algorithm terminate at iteration $t\leq t_{\max}$. Then, $\hat{D}^*_{\mathcal{W}_t}=\hat{D}_{\mathcal{W}_t}\left(\hat{\alpha}^{(t)}\right)=\hat{D}_{\mathcal{W}_t}\left(\hat{\alpha}\right)$ and, by strong duality, $(\sum_{\mathbf{c}}\hat{\alpha}_{\mathbf{c}}\hat{\Psi}_{\mathbf{c}},\max_{\mathbf{c}\in\mathcal{W}_t}\xi^{(t)}_{\mathbf{c}})$ is optimal for the corresponding primal problem

OP 5.

$$\begin{aligned}
\min_{\mathbf{w},\ \xi\geq 0}\quad & P(\mathbf{w},\xi)=\frac{1}{2}\langle\mathbf{w},\mathbf{w}\rangle+C\xi\\
s.t.\quad & \frac{1}{m}\sum_{i=1}^m c_i-\left\langle\mathbf{w},\hat{\Psi}_{\mathbf{c}}\right\rangle\leq\xi,\qquad\forall\mathbf{c}\in\{0,1\}^m.
\end{aligned}$$

for $\hat{\Psi}_{\mathbf{c}}$ defined by (B1), i.e.

$$\begin{aligned}
\hat{D}_{\mathcal{W}_t}(\hat{\alpha}) &= \frac{1}{2}\left\langle\sum_{\mathbf{c}}\hat{\alpha}_{\mathbf{c}}\hat{\Psi}_{\mathbf{c}},\sum_{\mathbf{c}}\hat{\alpha}_{\mathbf{c}'}\hat{\Psi}_{\mathbf{c}'}\right\rangle+C\max_{\mathbf{c}\in\mathcal{W}_t}\xi^{(t)}_{\mathbf{c}}\\
&= \frac{1}{2}\sum_{\mathbf{c}\mathbf{c}'}\hat{\alpha}_{\mathbf{c}}\hat{\alpha}_{\mathbf{c}'}\hat{J}_{\mathbf{c}\mathbf{c}'}+C\max_{\mathbf{c}\in\mathcal{W}_t}\xi^{(t)}_{\mathbf{c}}
\end{aligned}$$

Separately, note that

$$\begin{aligned}\hat{D}_{\mathcal{W}_t}(\hat{\alpha}^{(t)}) - D_{\mathcal{W}_t}(\hat{\alpha}^{(t)}) &= \frac{1}{2}\left(\hat{\alpha}^{(t)}\right)^T \left(J_{\mathcal{W}_t} - \hat{J}_{\mathcal{W}_t}\right)\hat{\alpha}^{(t)} \\ &= -\frac{1}{2}\left(\hat{\alpha}^{(t)}\right)^T \delta\hat{J}_{\mathcal{W}_t}\hat{\alpha}^{(t)} \\ &\leq \frac{C^2}{2}\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_\sigma \end{aligned} \tag{B8}$$

Denote by $\bar{\alpha}$ the vector in $\mathbb{R}^{2^m}$ given by

$$\bar{\alpha}_{\mathbf{c}} = \begin{cases} \hat{\alpha}_{\mathbf{c}} & \mathbf{c} \in \mathcal{W}_t \\ 0 & \text{otherwise} \end{cases}$$

Then,

$$\begin{aligned} P(\hat{\mathbf{w}}, \hat{\xi}) - P(\mathbf{w}^*, \xi^*) &= P(\hat{\mathbf{w}}, \hat{\xi}) - D(\alpha^*) \\ &\leq P(\hat{\mathbf{w}}, \hat{\xi}) - D(\bar{\alpha}) \\ &= P(\hat{\mathbf{w}}, \hat{\xi}) - D_{\mathcal{W}_t}(\hat{\alpha}^{(t)}) \\ &\leq P(\hat{\mathbf{w}}, \hat{\xi}) - \hat{D}_{\mathcal{W}_t}(\hat{\alpha}^{(t)}) + \frac{C^2}{2}\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_\sigma \\ &= \frac{1}{2}\sum_{\mathbf{cc}'}\hat{\alpha}_{\mathbf{c}}\hat{\alpha}_{\mathbf{c}'}\left(\Psi_{\mathbf{c}}^T\Psi_{\mathbf{c}'} - \hat{\Psi}_{\mathbf{c}}^T\hat{\Psi}_{\mathbf{c}'}\right) + C\left(\hat{\xi} - \max_{\mathbf{c}\in\mathcal{W}_t}\xi_{\mathbf{c}}^{(t)}\right) + \frac{C^2}{2}\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_\sigma \\ &\leq \frac{C}{t_{\max}} + C^2\left\|\delta\hat{J}_{\mathcal{W}_{t_{\max}}}\right\|_\sigma \\ &\leq \frac{2C}{t_{\max}} \\ &= 2\min\left\{\frac{C\epsilon}{4}, \frac{\epsilon^2}{16R^2}\right\} \end{aligned}$$

The first inequality follows from the fact that $\bar{\alpha}$ is feasible for OP 4. The second inequality is due to (B8). The third comes from the definition $\hat{\xi} - \max_{\mathbf{c}\in\mathcal{W}_t}\xi_{\mathbf{c}}^{(t)} = \frac{1}{t_{\max}}$and observing that $\frac{1}{2}\sum_{\mathbf{cc}'}\hat{\alpha}_{\mathbf{c}}\hat{\alpha}_{\mathbf{c}'}\left(\Psi_{\mathbf{c}}^T\Psi_{\mathbf{c}'} - \hat{\Psi}_{\mathbf{c}}^T\hat{\Psi}_{\mathbf{c}'}\right) = -\frac{1}{2}\left(\hat{\alpha}^{(t)}\right)^T\delta\hat{J}_{\mathcal{W}_t}\hat{\alpha}^{(t)} \leq \frac{C^2}{2}\left\|\delta\hat{J}_{\mathcal{W}_t}\right\|_\sigma$, and the fourth inequality follows from Lemma 3.

□

Appendix C: Proof of Theorem 7

Theorem 7. *Algorithm* 2 *has a time complexity of*

$$\tilde{O}\left(\frac{CR^3\log(1/\delta)}{\Psi_{\min}}\left(\frac{t_{\max}^2}{\epsilon}\cdot m + t_{\max}^5\right)T_\Phi\right) \tag{2}$$

where $\Psi_{\min} = \min_{\mathbf{c}\in\mathcal{W}_{t_f}} \|\Psi_{\mathbf{c}}\|$*, and* $t_f \leq t_{\max}$ *is the iteration at which the algorithm terminates.*

Proof. The initial storing of data to qRAM take time $\tilde{O}(md)$. Thereafter, each iteration t involves

- $O(t^2)$ inner product estimations $IP_{\epsilon_J,\delta_J}(\Psi_{\mathbf{c}}, \Psi_{\mathbf{c}'})$, for all pairs $\mathbf{c}, \mathbf{c}' \in \mathcal{W}_t$. By Theorem 4, each requires time $\tilde{O}\left(\frac{\log(1/\delta_J)}{\epsilon_J}\frac{R^3}{\min\{\|\Psi_{\mathbf{c}},\|\Psi_{\mathbf{c}'}\|\|\}}T_\Phi\right)$. By design $\epsilon_J = \frac{1}{Ctt_{max}} \geq \frac{1}{Ct_{\max}^2}$ and $\delta_J = \frac{\delta}{2t^2\, t_{\max}} \geq \frac{\delta}{2t_{\max}^3}$. The running time to compute all t^2 inner products is therefore $\tilde{O}\left(\frac{CR^3}{\Psi_{\min}}\log\left(\frac{2t_{\max}^3}{\delta}\right)t_{\max}^4 T_\Phi\right)$.

- The classical projection of a $t \times t$ matrix onto the p.s.d cone, and a classical optimization subroutine to find $\hat{\alpha}$. These take time $O(t^3)$ and $O(t^4)$ respectively, independent of m.

- Storing the $\hat{\alpha}_{\mathbf{c}}$ for $\mathbf{c} \in \mathcal{W}_t$ and the $\frac{c_i^{(t+1)}y_i}{\sqrt{m}}\|\Phi(\mathbf{x}_i)\|$ for $i = 1, \ldots, m$ in qRAM, and computing $\eta_{\mathbf{c}}$ classically. These take time $\tilde{O}(t_{\max})$, $\tilde{O}(m)$ and $O(m)$ respectively.

- m inner product estimations $IP_{\epsilon,\delta_\zeta}\left(\sum_{\mathbf{c}\in\mathcal{W}^t}\alpha_{\mathbf{c}}^{(t)}\Psi_{\mathbf{c}}, y_i\Phi(\mathbf{x}_i)\right)$, for $i = 1, \ldots, m$. By Theorem 5, each of these can be estimated to accuracy ϵ with probability at least $1 - \frac{\delta}{2m\, t_{\max}}$ in time $\tilde{O}\left(\frac{C|\mathcal{W}_t|}{\epsilon}\log\left(\frac{2m|\mathcal{W}_t|\, t_{\max}}{\delta}\right)\frac{R^3}{\min_{\mathbf{c}\in\mathcal{W}_t}\|\Psi_{\mathbf{c}}\|}T_\Phi\right)$. As $|\mathcal{W}_t| \leq t_{\max}$, it follows that all m inner products can be estimated in time $\tilde{O}\left(\frac{CR^3}{\Psi_{\min}}\frac{mt_{\max}}{\epsilon}\log\left(\frac{1}{\delta}\right)T_\Phi\right)$.

The total time per iteration is therefore

$$\tilde{O}\left(\frac{CR^3\log(1/\delta)}{\Psi_{\min}}\left(t_{\max}^4 + \frac{mt_{\max}}{\epsilon}\right)T_\Phi\right)$$

and since the algorithm terminates after at most $t_{\max}$ steps, the result follows.

□

Appendix D: Proof of Equation (B5)

Let $D(\alpha) = -\frac{1}{2}\alpha^T J\alpha + c^T\alpha$ where J is positive semi-definite. Here we show that

$$\max_{0\leq\beta\leq C}\left(D(\alpha+\beta\eta) - D(\alpha)\right) \geq \frac{1}{2}\min\left\{C, \frac{\eta^T\nabla D(\alpha)}{\eta^T J\eta}\right\}\eta^T\nabla D(\alpha)$$

for any η satisfying $\eta^T\nabla D(\alpha) > 0$. The change in D under a displacement $\beta\eta$ for some $\beta \geq 0$ satisfies

$$\delta D \stackrel{\text{def}}{=} D(\alpha+\beta\eta) - D(\alpha) = \beta\eta^T\nabla D(\alpha) - \frac{\beta^2}{2}\eta^T J\eta$$

which is maximized when

$$\frac{\partial}{\partial\beta}\delta D = \eta^T \nabla D(\alpha) - \beta\, \eta^T J \eta = 0$$
$$\Rightarrow \beta^* = \frac{\eta^T \nabla D(\alpha)}{\eta^T J \eta}$$

If $\beta^* \leq C$ then $\delta D = \frac{1}{2}\frac{\left(\eta^T \nabla D(\alpha)\right)^2}{\eta^T J \eta}$. If $\beta^* > C$ then, as D is concave, the best one can do is choose $\beta = C$, which gives

$$\begin{aligned}\delta D &= C\eta^T \nabla D(\alpha) - \frac{C^2}{2}\eta^T J \eta \\ &\geq \frac{C}{2}\eta^T \nabla D(\alpha)\end{aligned}$$

where the last line follows from $\beta^* > C \Rightarrow \eta^T \nabla D(\alpha) = \beta^* \eta^T J \eta \geq C\eta^T J \eta$.

Appendix E: Choice of quantum feature map

Let Z_j be the Pauli Z operator acting on qubit j in an N qubit system. Define $M = \frac{1}{N}\left(\sum_j Z_j\right)^2$ and let $M = \sum_{\mathbf{c},\mathbf{c}' \in \{0,1\}^N} M_{\mathbf{c}\mathbf{c}'} |\mathbf{c}\rangle\langle\mathbf{c}'|$ be M expressed in the computational basis. Define the vectorized form of M to be $|M\rangle = \sum_{\mathbf{c},\mathbf{c}' \in \{0,1\}^N} M_{\mathbf{c}\mathbf{c}'} |\mathbf{c}\rangle |\mathbf{c}'\rangle$. Define the states

$$|W\rangle \propto |0\rangle |M\rangle - \mu_0 |1\rangle |\mathbf{0}\rangle$$
$$|\Psi\rangle = \frac{|0\rangle |\psi\rangle |\psi\rangle + |1\rangle |\mathbf{0}\rangle}{\sqrt{2}}$$

where $\mu_0 > 0$, $|\psi\rangle$ is any N qubit state, and $|\mathbf{0}\rangle$ is the all zero state on $2N$ qubits. It holds that

$$\begin{aligned}\langle W|\Psi\rangle &\propto \langle M|\psi\rangle |\psi\rangle - \mu_0 \\ &= \langle\psi|M|\psi\rangle - \mu_0\end{aligned}$$

Thus

$$\langle W|\Psi\rangle \begin{cases} \geq 0 & \langle\psi|M|\psi\rangle \geq \mu_0 \\ < 0 & \langle\psi|M|\psi\rangle < \mu_0 \end{cases}$$

[1] Thorsten Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226. ACM, 2006.

[2] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[3] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[4] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[5] Michael C Ferris and Todd S Munson. Interior-point methods for massive support vector machines. *SIAM Journal on Optimization*, 13(3):783–804, 2002.

[6] Olvi L Mangasarian and David R Musicant. Lagrangian support vector machines. *Journal of Machine Learning Research*, 1(Mar):161–177, 2001.

[7] S Sathiya Keerthi and Dennis DeCoste. A modified finite newton method for fast solution of large scale linear svms. *Journal of Machine Learning Research*, 6(Mar):341–361, 2005.

[8] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[9] Thorsten Joachims. Making large-scale svm learning practical. In *Advances in Kernel Methods-Support Vector Learning*. MIT-press, 1999.

[10] John C Platt. *Fast training of support vector machines using sequential minimal optimization*. MIT press, 1999.

[11] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):27, 2011.

[12] Ronan Collobert and Samy Bengio. Svmtorch: Support vector machines for large-scale regression problems. *Journal of machine learning research*, 1(Feb):143–160, 2001.

[13] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

[14] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Physical review letters*, 113(13):130503, 2014.

[15] Vojtěch Havlíček, Antonio D Córcoles, Kristan Temme, Aram W Harrow, Abhinav Kandala, Jerry M Chow, and Jay M Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209–212, 2019.

[16] Maria Schuld and Nathan Killoran. Quantum machine learning in feature hilbert spaces. *Physical review letters*, 122(4):040504, 2019.

[17] Iordanis Kerenidis, Anupam Prakash, and Dániel Szilágyi. Quantum algorithms for second-order cone programming and support vector machines. *arXiv preprint arXiv:1908.06720*, 2019.

[18] Tomasz Arodz and Seyran Saeedi. Quantum sparse support vector machines. *arXiv preprint arXiv:1902.01879*, 2019.

[19] Tongyang Li, Shouvanik Chakrabarti, and Xiaodi Wu. Sublinear quantum algorithms for training linear and kernel-based classifiers. *arXiv preprint arXiv:1904.02276*, 2019.

[20] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of machine learning research*, 6(Sep):1453–1484, 2005.

[21] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum random access memory. *Physical review letters*, 100(16):160501, 2008.

[22] Anupam Prakash. *Quantum algorithms for linear algebra and machine learning.* PhD thesis, UC Berkeley, 2014.

[23] Ewin Tang. A quantum-inspired classical algorithm for recommendation systems. *arXiv preprint arXiv:1807.04271*, 2018.

[24] Ewin Tang. Quantum-inspired classical algorithms for principal component analysis and supervised clustering. *arXiv preprint arXiv:1811.00414*, 2018.

[25] András Gilyén, Seth Lloyd, and Ewin Tang. Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension. *arXiv preprint arXiv:1811.04909*, 2018.

[26] Juan Miguel Arrazola, Alain Delgado, Bhaskar Roy Bardhan, and Seth Lloyd. Quantum-inspired algorithms in practice. *arXiv preprint arXiv:1905.10415*, 2019.

[27] Stephen Boyd and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[28] Jonathan Allcock, Chang-Yu Hsieh, Iordanis Kerenidis, and Shengyu Zhang. Quantum algorithms for feedforward neural networks. *arXiv preprint arXiv:1812.03089*, 2018.

[29] Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 67. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[30] Mario Motta, Chong Sun, Adrian Teck Keng Tan, Matthew JO' Rourke, Erika Ye, Austin J Minnich, Fernando GSL Brandao, and Garnet Kin Chan. Quantum imaginary time evolution, quantum lanczos, and quantum thermal averaging. *arXiv preprint arXiv:1901.07653*, 2019.

[31] Chang-Yu Hsieh, Qiming Sun, Shengyu Zhang, and Chee Kong Lee. Unitary-coupled restricted boltzmann machine ansatz for quantum simulations. *arxiv prepring quant-ph/1912.02988*, 2019.

[32] Jonathan Romero, Jonathan P Olson, and Alan Aspuru-Guzik. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology*, 2(4):045001, 2017.

[33] Edward Farhi and Hartmut Neven. Classification with quantum neural networks on near term processors. *arXiv preprint arXiv:1802.06002*, 2018.

[34] Seth Lloyd, Maria Schuld, Aroosa Ijaz, Josh Isaac, and Nathan Killoran. Quantum embeddings for machine learning. *arXiv preprint arXiv:2001.03622*, 2020.

[35] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74, 2002.