Quantum Support Vector Machines

Iordanis Kerenidis [1], Anupam Prakash [1] and Dániel Szilágyi[1]

[1]CNRS, IRIF, Université Paris Diderot, Paris, France.

June 28, 2019

Quantum Computing

- Quantum states (qudits) are unit vectors in Hilbert space $\mathbb{C}^d$.

$$|\phi\rangle = \sum_{i \in [d]} \beta_i |i\rangle .$$

- The measurement M in the standard yields a probabilistic outcome,

$$\Pr[M(|\phi\rangle) = i] = |\beta_i|^2.$$

- Multi-qudit quantum systems are represented by a tensor product $|\phi_1, \phi_2\rangle$.
- Quantum computers can apply unitary operations to states and perform measurements.
- Hadamard gate:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

Quantum Computing

- It is easy to create exponential sized superpositions,

$$H^n |0^n\rangle = \frac{1}{\sqrt{2^n}} \sum_{i \in [2^n]} |i\rangle .$$

- What speedups do quantum computers offer over the best known classical algorithms?
- Exponential speedups: Integer factoring, discrete logarithms [Shor], sampling from solutions to structured sparse linear systems. [Harrow, Hassidim and Lloyd].
- Quadratic speedups: Finding a marked element in a database of N items in time $O(\sqrt{N})$ [Grover].
- Significant polynomial speedups: Recommendation systems [KP16], quantum machine learning, quantum optimization?

Quantum Machine Learning

- Input encoding: How to encode a classical vector $x \in \mathbb{R}^n$ into quantum state? How to encode matrices $A \in \mathbb{R}^{n \times n}$?
- Quantum linear algebra: Given encodings, there are efficient quantum linear algebra algorithms to obtain states $|Ax\rangle, |A^{-1}x\rangle$ and $|\Pi_A x\rangle$ where $\Pi_A(x)$ is the projection of x onto $Col(A)$.
- Output extraction: How to obtain classical information from the quantum state? (i) Measure in standard basis to sample. (ii) Perform quantum state tomography with ℓ_∞ or ℓ_2 norm guarantees.

QRAM data structures

- QRAM (Quantum Random Access Memory) is a powerful memory model for quantum access to arbitrary datasets.
- Given $x_i, i \in [N]$ stored in the QRAM, the following queries require time polylog(N),

$$|i, 0\rangle \rightarrow |i, x_i\rangle$$

- Weaker quantum memory models are applicable only for well-structured datasets and are not suitable for general ML problems.

Definition

A QRAM data structure for storing a dataset D of size N is efficient if it can be constructed in a single pass over the entries (i, d_i) for $i \in [N]$ and the update time per entry is $O(\text{polylog}(N))$.

Input Encodings

- Encoding vectors: There are efficient QRAM data structures for storing vector $v \in \mathbb{R}^n$ that allow $|v\rangle$ to be prepared in time $O(\log^2 n)$.
- Encoding matrices: A matrix $A \in \mathbb{R}^{n \times n}$ is encoded as a unitary block encoding, that is,

$$U_A = \begin{pmatrix} A/\mu(A) & . \\ . & . \end{pmatrix}$$

.

- How to construct block encodings for A and what $\mu(A)$ can be achieved?
- The optimal value of $\mu(A) \geq \|A\|$, any minor of a unitary matrix has spectral norm at most 1.

Input Encodings

- For quantum linear algebra, it is standard to normalize so that $\|A\| = 1$.

Theorem (KP16, KP17)

There are efficient QRAM data structures for storing $A \in \mathbb{R}^{n \times n}$, such that with access to these data structures a block encoding for A with $\mu(A) \leq \sqrt{n}$ can be implemented in time $O(polylog(n))$.

- We note that $\mu(A) < \sqrt{n}$ can be much less than $O(\sqrt{n})$ for low rank matrices and matrices with bounded ℓ_1 norms for rows/columns.

Quantum linear algebra

- Let $\kappa(A) = \lambda_{max}(A)/\lambda_{\min}(A)$ be the condition number of matrix A.
- Given efficient block encodings for A, there are efficient quantum linear algebra procedures. [KP16, KP17, CGJ18].
- Theorem: A state ϵ-close to $|Ax\rangle$ or $|A^{-1}x\rangle$ in the ℓ_2 norm can be generated in time $O(\kappa(A)\mu(A)\log(1/\epsilon))$.
- Theorem: The norm $\|Ax\|$ or $\|A^{-1}x\|$ can be estimated to relative error ϵ in time $O(\frac{\kappa(A)\mu(A)}{\epsilon}\log(1/\epsilon))$.
- As μ is sublinear, quantum linear algebra provides large gains in efficiency over the classical $O(n^3)$ for many classes of matrices.

Output extraction

- The quantum states $|A^{-1}x\rangle$ are not the same as the output for classical linear system solvers.
- If we measure $|A^{-1}x\rangle$ in the standard basis, we obtain a sample from the squared ℓ_2 distribution for the state. [Recommendation systems].
- Using Chernoff bounds, with $O(1/\epsilon^2)$ samples we can recover an approximation $\|x - \tilde{x}\|_\infty \leq \epsilon$.
- There is an ℓ_2-tomography algorithm with $O(n \log n/\epsilon^2)$ and approximation $\|x - \tilde{x}\|_2 \leq \epsilon$. [KP18].
- The ℓ_2 tomography algorithm is used for quantum optimization using the interior point method.

Interior Point Method overview

- Interior point methods are widely used for solving Linear programs (LP), Second Order Cone Programs (SOCP) and Semidefinite Programs (SDP).
- Running time for SDP algorithms will be given in terms of dimension n, number of constraints m and error ϵ.
- The classical IPM starts with feasible solutions (S, Y) to the optimization problem and updates them $(S, Y) \rightarrow (S + dS, Y + dY)$ iteratively.
- The updates (dS, dY) are obtained by solving a $O(n + m)$ dimensional linear system called the Newton linear system.
- After $O(\sqrt{n} \log(1/\epsilon))$ iterations, the method converges to feasible solutions (S, Y) with duality gap at most ϵ, that is solutions are ϵ close to the optimal.

Quantum SDP algorithms

- Does quantum linear algebra offer speedups for optimization using IPMs?
- Quantum SDP algorithms using multiplicative weights method were proposed recently [Brandao-Svore 17].
- After many improvements, the best running time for a quantum SDP algorithm [AG19] using this framework is,

$$\tilde{O}\left(\left(\sqrt{m}+\sqrt{n}\left(\frac{Rr}{\epsilon}\right)\right)\left(\frac{Rr}{\epsilon}\right)^{4}\sqrt{n}\right).$$

- For Max-Cut and scheduling LPs , the complexity is at least $O(n^6)$ [AGGW17, Theorem 20].

Quantum SDP algorithms

- We provided a quantum interior point method with complexity $\widetilde{O}(\frac{n^{2.5}}{\xi^2}\mu\kappa^3\log(1/\epsilon))$ for SDPs and $\widetilde{O}(\frac{n^{1.5}}{\xi^2}\mu\kappa^3\log(1/\epsilon))$ for LPs . [KP18].
- The output of our algorithm is a pair of matrices (S, Y) that are ϵ-optimal ξ-approximate SDP solutions.
- The parameter μ is at most $\sqrt{2}n$ for SDPs and $\sqrt{2n}$ for LPs .
- The parameter κ is an upper bound on the condition number of the intermediate solution matrices.
- If the intermediate matrices are 'well conditioned', the running time scales as $\widetilde{O}(n^{3.5})$ and $\widetilde{O}(n^2)$.
- Does this provide speedups in practice?

Second Order Cone Programs

- The SOCP is an optimization problem over the product of Lorentz cones $\mathcal{L}_k$,

$$\mathcal{L}^k = \left\{ \boldsymbol{x} = (x_0; \widetilde{\boldsymbol{x}}) \in \mathbb{R}^k \;\middle|\; \|\widetilde{\boldsymbol{x}}\| \leq x_0 \right\}.$$

- The standard form of the SOCP is the following optimization problem:

$$\begin{array}{lll} \min\limits_{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_r} & \boldsymbol{c}_1^T \boldsymbol{x}_1 + \cdots \boldsymbol{c}_r^T \boldsymbol{x}_r & \\ \text{s.t.} & A^{(1)} \boldsymbol{x}_1 + \cdots + A^{(r)} \boldsymbol{x}_r = \boldsymbol{b} & (1) \\ & \boldsymbol{x}_i \in \mathcal{L}^{n_i}, \ \forall i \in [r], & \end{array}$$

- The rank r is like the number of constraints while n is the dimension of the solution vector, classical IPM for SOCP has complexity $O\left(\sqrt{r} n^{\omega} \log(n/\epsilon)\right)$.

Quantum IPM for SOCP

- Starts with initial feasible solution (x, s, y) for primal-dual SOCP pair and solves the (*Newton system*) to compute the updates $(\Delta\boldsymbol{x}, \Delta\boldsymbol{y}, \Delta\boldsymbol{s})$:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ \mathrm{Arw}(\boldsymbol{s}) & 0 & \mathrm{Arw}(\boldsymbol{x}) \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{x} \\ \Delta\boldsymbol{y} \\ \Delta\boldsymbol{s} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b} - A\boldsymbol{x} \\ \boldsymbol{c} - \boldsymbol{s} - A^T\boldsymbol{y} \\ \sigma\mu\boldsymbol{e} - \boldsymbol{x} \circ \boldsymbol{s} \end{bmatrix}. \quad (2)$$

- The Newton linear system is much simpler than case of general SDPs.
- Converges in $O(\sqrt{r}\log(1/\epsilon))$ iterations like the classical algorithm. General analysis using Euclidean Jordan algebras.

Quantum IPM for SOCP

- There is a quantum Algorithm that outputs a solution $x_i \in \mathcal{L}^{n_i}$ that achieves an objective value that is within ϵ of the optimal value in time,

$$T = \widetilde{O}\left(\sqrt{r}\log\left(\mu_0/\epsilon\right) \cdot \frac{n\kappa\zeta}{\delta^2}\log\left(\frac{\kappa\zeta}{\delta}\right)\right).$$

- $\zeta \leq \sqrt{n}$ is a factor that appears in quantum linear system solvers.
- κ is an upper bound on the condition number of the matrices arising in the interior point method for SOCPs.
- The parameter δ is a lower bound on how close are the intermediate iterates to the boundaries of the respective cones.
- How does this perform in practice?

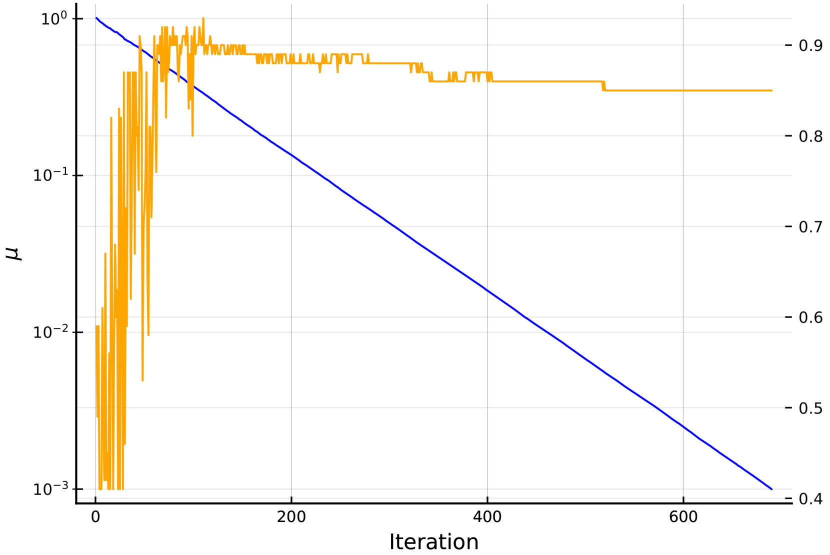
Support Vector Machines

- The ℓ_1-regularized SVM for m data points of dimension n is the following optimization problem,

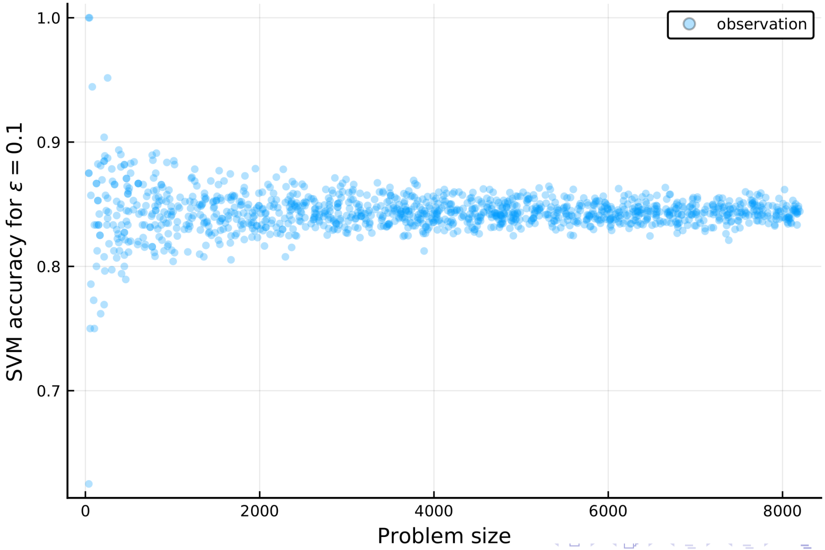
$$\begin{aligned} \min_{\boldsymbol{w}, b, \boldsymbol{\xi}} \quad & \|\boldsymbol{w}\|^2 + C\,\|\boldsymbol{\xi}\|_1 \\ \text{s.t.} \quad & y^{(i)}(\boldsymbol{w}^T \boldsymbol{x}^{(i)} + b) \geq 1 - \xi_i, \ \forall i \in [m] \\ & \boldsymbol{\xi} \geq 0. \end{aligned} \tag{3}$$

- If $\boldsymbol{t} = (t+1; t; \boldsymbol{w})$ is in the Lorentz cone, then $2t + 1 > \|w\|^2$, the norm constraint becomes linear in t.
- The ℓ_1-SVM reduces to an instance of SOCP with rank $2m + 4$ constraints and dimension $3m + 2n + 7$.
- Experiments on random SVMs: Generate data points and separating hyperplane uniformly at random from $[-1, 1]^n$. Flip a p fraction of the labels. Shift by direction sampled from $N(0, 2I)$.

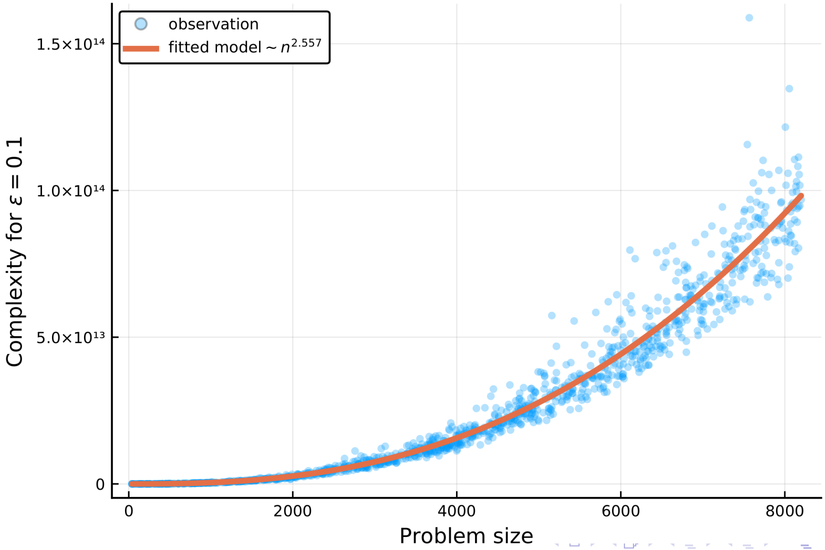
Experimental results – accuracy and duality gap



Experimental results – accuracy with problem size



Experimental results – asymptotoc speedup



Conclusions

- The quantum SVM algorithm achieves an asymptotic speedup on random SVM instances with running time $O(n^{2.557})$ as opposed to the classical IPM with running time $O(n^{3.5})$.
- This also indicates the potential for similar asymptotic speedups using quantum optimization for problems relevant in practice.