Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning

Han-Hsuan Lin

NTHU

February 18, 2021

Joint work with Nai-Hui Chia, Andras Gilyen, Tongyang Li, Ewin Tang, and Chunhao Wang

Intro

- There were quantum machine learning (QML) algorithm. (Quantum algorithms that solve classical machine learning problems.)
- In this work, we dequantized a large number of known QML algorithms. (Giving classical algorithms whose runtime is polynomial compared to the runtime of QML algorithms, showing that the QML algorithms cannot have exponential speedup.)

Overview

History Quantum Machine Learning

Quantum States

Recall that in quantum mechanics, data are stored as quantum states.

Two properties of a quantum state.

1. A (pure) quantum state is a complex unit vector.

   Ex: $\begin{bmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} i \end{bmatrix}$

2. When we measure a quantum state, we get a random component with probability proportional to its amplitude squared.

   Ex: $\begin{bmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} i \end{bmatrix} \xrightarrow{\textit{measure}} \hat{e}_0$ with probability $\frac{1}{4}$, $\hat{e}_1$ with probability $\frac{3}{4}$

Problem: Solve $Ax = b$

A is a sparse matrix. b is a unit vector. A and b are given, solve for x.

- HHL quantum algorithm: Given b as a quantum state, get $x/|x|$ as a quantum state , in time polylog of the size of b.
- For a classical algorithm, even reading b would take linear time. "Exponential speedup".
- Using HHL, one can therefore calculate $\langle x| M |x\rangle$ efficiently for some operator M.
- HHL uses phase estimation to manipulate eigenvalues and eigenvectors of A.

Data Issue of HHL

b as a quantum state: the algorithm has access to a copy of quantum state $|b\rangle$ such that $\langle i|b\rangle = b_i$.

Issue: how do we get $|b\rangle$

QML following HHL

- We didn't really have an answer to the data issue, but that didn't stop people from publishing QML papers.
- Various QML algorithm for different problems are proposed with similar ideas of HHL. These QML algorithm need to take some input data as a quantum state, therefore having the same data issue as HHL.
- Examples: semi-definite programming (SDP), Hamiltonian simulation, supervised clustering, principal component analysis, support vector machine, and discriminant analysis, recommendation system.

[KP17] algorithm for recommendation system

Finally, in [KP17], the authors solved the data issue by showing how to compute the quantum state needed from classical inputs with pre-computation. (The pre-computation takes linear time, but such pre-computation makes sense in the context of recommendation system.)

"Dequantization": Classical Sampling Techniques

Ewin Tang's dequantization

- Ewin Tang [Tan18a] dequantized [KP17]'s quantum algorithm for recommendation system.
- Key observation: if we have a quantum state, we can measure it and sample from it, with basically no quantum computation.

Reminder: Quantum States

Two facts about quantum state.

1. A (pure) quantum state is a complex unit vector.

   Ex: $\begin{bmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} i \end{bmatrix}$

2. When we measure a quantum state, we get a random component with probability proportional to it's amplitude squared.

   Ex: $\begin{bmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} i \end{bmatrix} \xrightarrow{measure} \hat{e}_0$ with probability $\frac{1}{4}$, $\hat{e}_1$ with probability $\frac{3}{4}$

FKV sampling

There exists work in classical literature about how to construct fast algorithms given the ability to do sampling.

In particular, sampling techniques from [FKV04] gives a succinct approximation of a low rank matrix.

We build on this technique to get approximation to singular value decomposition and singular value transformation of a low rank matrix.

Sampling access of a matrix

Given a matrix A, we assume we have two sampling access

- sample a row index according to two-norms of each row.
- input a row index, sample a column index according to norm squared of elements of chosen row.

We can efficiently sample from any vector if we pre-compute a binary tree.

$$\|M(i,\cdot)\|^2$$

$$|M(i,1)|^2 + |M(i,2)|^2 \qquad |M(i,3)|^2 + |M(i,4)|^2$$

$$\big(|M(i,1)|^2, M(i,1)\big) \quad \big(|M(i,2)|^2, M(i,2)\big) \quad \big(|M(i,3)|^2, M(i,3)\big) \quad \big(|M(i,4)|^2, M(i,4)\big)$$

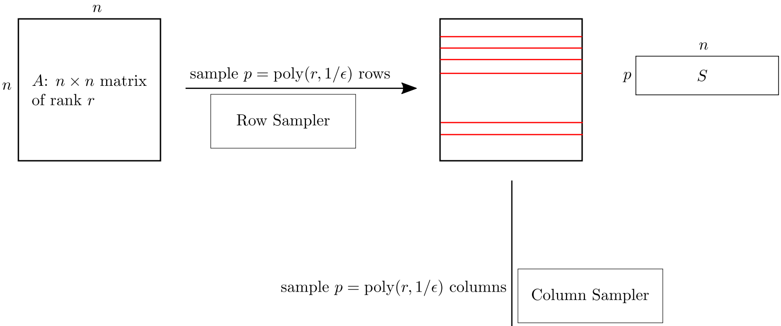
Note: Similar construction is used in [KP17] (QRAM)

FKV approximation

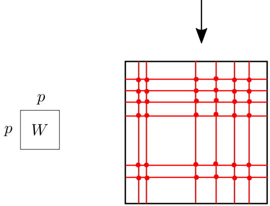
How to find an "approximation" of a rank r, $n \times n$ matrix A. ($r \ll n$)

- Intuition: Because A is low rank, just a few rows of A will span the whole row space of A.
- Sample $p = O(\mathrm{poly}(r))$ rows from A. Do some normalization to get $p \times n$ matrix S. $S^\dagger S \approx A^\dagger A$.
- Similarly, sample p columns from S to get $p \times p$ matrix W. $WW^\dagger \approx SS^\dagger$.
- W is small ($p \times p$) so we can just write it down and do singular value decomposition (SVD).
- A, S, and W all have the same singular values and related singular vectors. Use SVD of W to "normalize" $S^\dagger$ to get projector V.
- $A \approx AVV^\dagger$

FKV approximation



Sample a small submatrix $A \to S \to W$



FKV approximation

Compute SVD of W:

$$W = \sum_{i=1}^{p} \sigma_i u_i u_i^\dagger$$

Define vector v_i:

$$v_i := \frac{1}{\sigma_i} S^\dagger u_i$$

Define the $n \times p$ matrix V:

$$V := \begin{array}{cc} \begin{bmatrix} v_1 & \cdots & v_p \end{bmatrix} & n \\ p & \end{array}$$

[FKV04]: $\|A - AVV^\dagger\|_F \leq \epsilon$

Succinct description of V?

- Each column of V is a linear combination of the sampled rows of A. Record V efficiently by writing down the linear coefficients and which rows are sampled.
- We can calculate lots of things about V efficiently from the succinct description.

Sampling Toolbox

1. Can sample from linear combination of rows of A by rejection sampling.
2. By (1), can sample from columns of V.
3. If we can sample from two vectors x and y, we can calculate $x^\dagger y$. (By estimating a random variable that distributed with probability $|x_i|^2$ and have values y_i/x_i)
4. If we can sample from matrix B, vectors x, y, we can calculate $x^\dagger By$. [CLW18]
5. If we can sample from matrix B and query entries of matrix C, we can calculate $\mathrm{Tr}[BC]$.[GLT18]

The Framework: Singular Value Transformation

Our breakthrough: getting singular value decomposition

Recall: The FKV gives us V such that $A \approx AVV^\dagger$.

Objective: get singular value decomposition of A.

Solution:

- If we can get V such that $A \approx AVV^\dagger$ by sampling rows, then we can get U such that $A \approx UU^\dagger A$ by sampling columns. (Double the overhead in pre-computation to get sampling access to columns.)
- $A \approx UU^\dagger AVV^\dagger$.
- Note that $UU^\dagger AVV^\dagger = U(U^\dagger AV)V^\dagger$
- $U^\dagger AV$ is small, so we can calculate every entry of it and write it down. (we can calculate $x^\dagger By$)
- singular value decomposition: $U^\dagger AV = U'DV'^\dagger$. Write down U', D, V'.
- Get $A \approx (UU')D(VV')^\dagger$, approximate singular decomposition of A.
- We have explicit description of D and succinct description of (UU') and (VV').

Singular Value Transformation

Singular value transformation: apply a function on singular values. I.e. If $A = UDV^\dagger$ and f is a real valued function, $f^{(SV)}(A) = U f(D) V^\dagger$.

Recall that we already get $A \approx (UU')D(VV')^\dagger$, so we can get approximation to $f^{(SV)}(A)$ by calculating $f(D)$.

By [GSLW19], many quantum machine learning algorithm can be described in terms of singular value transformation.

Applications

Matrix inversion

Problem: given low rank matrix A and vector b, calculate $x = A^{-1}b$.

Solution: apply singular value transformation to A with $f(x) = 1/x$. (With some modifications to cut off the singularity.)

Note: incomparable to the HHL algorithm, since HHL works on sparse matrices, while our classical algorithm need low rank matrices. But recall that HHL algorithm has data issue.

Solving low-rank Semi-Definite Programming (SDP)

This problem is parameterized by set of low rank matrices $\{A_i\}$. By the Matrix Multiplicative Weight method, we can solve SDP if we can calculate

$$\mathrm{Tr}[A_i \exp[-\varepsilon(A_{j_1} + A_{j_2} + \dots)]]$$

Solution: apply singular value transformation with $f(x) = e^{-\epsilon x}$ to $(A_{j_1} + A_{j_2} + \dots)$. (Actually this is a transformation on eigenvalues, eigenvalue transformation can be done similarly for Hermitian matrices.) (Need to combine sampling access to different A_j too.)

Hamiltonian simulation

Problem: given matrix H, unit vector ψ and positive number t, calculate $\exp(-iHt)\psi$.

Solution: apply singular value transformation to H with $f(x) = e^{-ixt}$. (Actually this is a transformation on eigenvalues.)

Conclusion

- We give a classical framework to apply functions on low rank matrices that runs in time polylog in dimension of the matrices.
- The framework applies singular value transformation to matrices.
- The framework gives classical algorithms of matching run time to quantum machine learning algorithms. Including problems like recommendation system, low rank matrix inversion, low rank SDP, Hamiltonian simulation, supervised clustering, principal component analysis , support vector machine, and discriminant analysis.

Hope for QML

- QML algorithms can still have polynomial speedup.
- QML algorithms can have exponential speed up if they can get quantum input naturally. (E.g. electromagnetic scattering.)

Thanks!

Recommendation System

Problem: Given a matrix A with small Frobenius norm, calculate the low rank approximation of A, A_k.

Solution: apply singular value transformation to A with a threshold function cutting off smaller eigenvalues. (The framework actually work as long as the Frobenius norm of A is small.)

Scott Aaronson. Shadow tomography of quantum states. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing*, pages 325–338. ACM, 2018, arXiv:1711.01053.

Ittai Abraham, Shiri Chechik, and Sebastian Krinninger. Fully dynamic all-pairs shortest paths with worst-case update-time revisited. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 440–452. SIAM, 2017, arXiv:1607.05132.

Joran van Apeldoorn and András Gilyén. Improvements in quantum SDP-solving with applications. 2018, arXiv:1804.05058.

Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-solvers: Better upper and lower bounds. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2017, arXiv:1705.01843.

Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

Alexandr Andoni, Piotr Indyk, and Ilya Razenshteyn. Approximate nearest neighbor search in high dimensions. 2018, arXiv:1806.09823.

Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 227–236. ACM, 2007.

Kurt M. Anstreicher. The volumetric barrier for semidefinite programming. *Mathematics of Operations Research*, 25(3):365–380, 2000.

Sepehr Assadi, Krzysztof Onak, Baruch Schieber, and Shay Solomon. Fully dynamic maximal independent set with sublinear update time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 815–826. ACM, 2018, arXiv:1802.09709.

Zeyuan Allen-Zhu, Yin Tat Lee, and Lorenzo Orecchia. Using optimization to obtain a width-independent, parallel, simpler, and faster positive sdp solver. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1824–1831.

Society for Industrial and Applied Mathematics, 2016, arXiv:1507.02259.

Sayan Bhattacharya, Deeparnab Chakrabarty, Monika Henzinger, and Danupon Nanongkai. Dynamic algorithms for graph coloring. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1–20. Society for Industrial and Applied Mathematics, 2018, arXiv:1711.04355.

Sayan Bhattacharya, Monika Henzinger, and Danupon Nanongkai. Fully dynamic approximate maximum matching and minimum vertex cover in $o(\log^3 n)$ worst case update time. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 470–489. SIAM, 2017, arXiv:1704.02844.

Fernando G. S. L. Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta M. Svore, and Xiaodi Wu. Quantum SDP solvers: Large speed-ups, optimality, and applications to quantum learning. 2017, arXiv:1710.02581.

Fernando G. S. L. Brandão and Krysta Svore. Quantum speed-ups for semidefinite programming. In *Proceedings of the 58th Annual IEEE*

Symposium on Foundations of Computer Science. IEEE, 2017, arXiv:1609.05537.

Timothy M. Chan, Kasper Green Larsen, and Mihai Pătrașcu. Orthogonal range searching on the RAM, revisited. In *Proceedings of the 27th Annual Symposium on Computational Geometry*, pages 1–10. ACM, 2011, arXiv:1103.5510.

Michael B. Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing*, ACM, 2019, arXiv:1810.07896.

Nai-Hui Chia, Han-Hsuan Lin, and Chunhao Wang. Quantum-inspired sublinear classical algorithms for solving low-rank linear systems. 2018, arXiv:1811.04852.

Xiao-Wen Chang, Christopher C. Paige, and G.W. Stewart. New perturbation analyses for the Cholesky factorization. *IMA Journal of Numerical Analysis*, 16(4):457–484, 1996.

- Alan Frieze, Ravi Kannan, and Santosh Vempala. Fast Monte-Carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51(6):1025–1041, 2004.
- Shmuel Friedland and Wasin So. On the product of matrix exponentials. *Linear Algebra and its Applications*, 196:193 – 205, 1994.
- Dan Garber and Elad Hazan. Approximating Semidefinite Programs in Sublinear Time. *Advances in Neural Information Processing Systems 24*, pages 1080–1088. Curran Associates, Inc., 2011.
- Dan Garber and Elad Hazan. Almost Optimal Sublinear Time Algorithm for Semidefinite Programming, 2012, arXiv:1208.5211.
- András Gilyén. Personal communication, 2019.
- Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

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

Gus Gutoski and Xiaodi Wu. Parallel approximation of min-max problems with applications to classical and quantum zero-sum games. In *Proceedings of the 27th Annual IEEE Symposium on Computational Complexity*, pages 21–31. IEEE, 2012.

Elad Hazan. Efficient algorithms for online convex optimization and their applications. PhD Thesis, Princeton University, 2006.

Monika R. Henzinger, Valerie King, and Valerie King. Randomized fully dynamic graph algorithms with polylogarithmic time per operation. *Journal of the ACM (JACM)*, 46(4):502–516, 1999.

Jacob Holm, Kristian de Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *Journal of the ACM (JACM)*, 48(4):723–760, 2001.

Rahul Jain and Penghui Yao. A parallel approximation algorithm for positive semidefinite programming. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*, pages 463–471. IEEE, 2011, arXiv:1104.2502.

Leonid G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):51–68, 1980.

Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems. In *Proceedings of the 8th Innovations in Theoretical Computer Science Conference*, pages 49:1–49:21, 2017, arXiv:1603.08675.

Iordanis Kerenidis and Anupam Prakash. A quantum interior point method for LPs and SDPs. 2018, arXiv:1808.09266.

Michael Luby and Noam Nisan. A parallel approximation algorithm for positive linear programming. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, pages 448–457. ACM, 1993.

James R. Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*. ACM, 2015, arXiv:1411.6317.

Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science*, pages 1049–1065. IEEE, 2015, arXiv:1508.04874.

Richard J. Lipton and Robert E. Tarjan. Applications of a planar separator theorem. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 162–170. IEEE, 1977.

John E. Mitchell. Polynomial interior point cutting plane methods. *Optimization Methods and Software*, 18(5):507–534, 2003.

Yurii Nesterov and Arkadi Nemirovsky. Conic formulation of a convex programming problem and duality. *Optimization Methods and Software*, 1(2):95–115, 1992.

Krzysztof Onak, Dana Ron, Michal Rosen, and Ronitt Rubinfeld. A near-optimal sublinear-time algorithm for approximating the minimum vertex cover size. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1123–1131. Society for Industrial and Applied Mathematics, 2012, arXiv:1110.1079.

David Poulin and Pawel Wocjan. Sampling from the thermal quantum Gibbs state and evaluating partition functions with a quantum computer. *Physical Review Letters*, 103(22):220502, 2009, arXiv:0905.2199.

Shay Solomon. Fully dynamic maximal matching in constant update time. In *Proceedings of the 57th Annual Symposium on Foundations of Computer Science*, pages 325–334. IEEE, 2016, arXiv:1604.08491.

Ewin Tang. A quantum-inspired classical algorithm for recommendation systems. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing*, ACM, 2019, arXiv:1807.04271.

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

Ewin Tang. Personal communication, 2019.

Mikkel Thorup. Worst-case update times for fully-dynamic all-pairs shortest paths. In *Proceedings of the 37th Annual ACM Symposium on Theory of Ccomputing*, pages 112–119. ACM, 2005.

Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

Xiaodi Wu. Parallelized solution to semidefinite programmings in quantum complexity theory, 2010, arXiv:1009.2211.

Drineas, P. and Kannan, R. and Mahoney, M.. Fast Monte Carlo Algorithms for Matrices I: Approximating Matrix Multiplication. In *SIAM Journal on Computing*, 36(1):132–157, 2006.

Harrow, Aram W., Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. Physical review letters 103.15 (2009): 150502.

Gilyén, A., Su, Y., Low, G. H., and Wiebe, N. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (pp. 193-204).