Quantum-inspired ℓ^2 sampling

and applications to machine learning

Faris Sbahi

3/5/19

Machine Learning

Quantum Machine Learning

Classical ℓ^2 sampling

Remarks

Today's talk

- In general, quantum machine learning algorithms convert quantum input states to the desired quantum output states.
- In practice, data is initially stored classically and the algorithm's output must be accessed classically as well.
- Today's focus: A practical way to make comparisons between classical and quantum algorithms is to analyze classical algorithms under ℓ^2 sampling conditions
- Tang: linear algebra problems in low-dimensional spaces (say constant or polylogarithmic) likely can be solved "efficiently" under these conditions
- Many of the initial practical applications of quantum machine learning were to problems of this type (e.g. Quantum Recommendation Systems - Kerendis, Prakash, 2016)

Machine Learning

Introduction

- Machine learning is a broad term for algorithms which are capable of finding patterns in data.
- Fundamental goal: capture these patterns in a "model" that *generalizes* to unseen data.
- These algorithms have two components:
  1. A learning element. Updates the model depending on its performance on the considered dataset.
  2. A performance element. Provides the measure of performance.
- Bottom line: "machine learning" is a somewhat hollow term. Many ML algorithms are in fact familiar linear algebraic techniques.

PCA

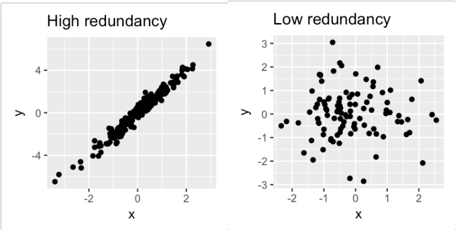
Motivation: Singular value transformation

- "Training" dataset $\mathcal{T}$ consists of the accessible samples of data. $\mathcal{T}$ is drawn from a subset of $\Omega \subset \mathbb{R}^d$ where each component represents a "feature".
- Samples from Ω are assumed to be drawn according to some distribution $\mathcal{D}$.

PCA

Motivation: Singular value transformation

- ”Training” dataset $\mathcal{T}$ consists of the accessible samples of data. $\mathcal{T}$ is drawn from a subset of $\Omega \subset \mathbb{R}^d$ where each component represents a ”feature”.
- Samples from Ω are assumed to be drawn according to some distribution $\mathcal{D}$.
- Example: data is collected on the heights and lengths of cherry blossom petals.



- How and why may it make sense to reduce the dimensionality of the feature space?

Moore-Penrose Pseudoinverse

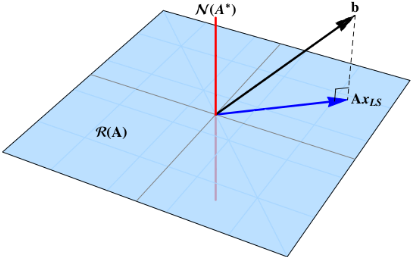
Motivation: Singular value transformation

- Let $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ unit vector. In machine learning, A is the matrix with rows given by the samples of $\mathcal{T}$.
- We wish to find the x_{LS} which satisfies $x_{LS} = \arg\min_x \|Ax - b\|_2$
- Notation: $x_{LS} = A^+ b$

Moore-Penrose Pseudoinverse

Motivation: Singular value transformation

- Let $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$ unit vector. In machine learning, A is the matrix with rows given by the samples of $\mathcal{T}$.
- We wish to find the x_{LS} which satisfies $x_{LS} = \arg\min_x \|Ax - b\|_2$
- Notation: $x_{LS} = A^+ b$
- Common strategy uses SVD: write $A = UDV^\dagger$ and then $A^+ = VD^+U^\dagger$ where D^+ simply inverts the non-zero diagonal entries.



Machine Learning

Quantum Machine Learning

Classical ℓ^2 sampling

Remarks

Moore-Penrose Pseudinverse (Quantum)

Harrow, Hassidim, Lloyd (orig.) Wiebe, Braun

- We can compute $A^+ |b\rangle = |x_{LS}\rangle$ in $\tilde{O}(log(N)(s^3\kappa^6)/\epsilon)$ time (query complexity)
- Uses a quantum algorithm based on phase estimation and Hamiltonian simulation
- Assumption: A is sparse with low condition number κ. Hamiltonian $(\hat{H})$ simulation is efficient when $\hat{H}$ is sparse. No low-rank assumptions are necessary.
- "Key" assumption: the quantum state $|b\rangle$ can be prepared efficiently.
- What happens if we assume low rank?

Machine Learning

Quantum Machine Learning

Classical ℓ^2 sampling

Remarks

In search of a ”fair” comparison

- How can we compare the speed of quantum algorithms with quantum input and quantum output to classical algorithms with classical input and classical output?
- Quantum machine learning algorithms can be exponentially faster than the best standard classical algorithms for similar tasks, but quantum algorithms get help through input state preparation.
- Want a practical classical model that helps its algorithms offer similar guarantees to quantum algorithms, while still ensuring that they can be run in nearly all circumstances one would run the quantum algorithm.

In search of a "fair" comparison

- How can we compare the speed of quantum algorithms with quantum input and quantum output to classical algorithms with classical input and classical output?
- Quantum machine learning algorithms can be exponentially faster than the best standard classical algorithms for similar tasks, but quantum algorithms get help through input state preparation.
- Want a practical classical model that helps its algorithms offer similar guarantees to quantum algorithms, while still ensuring that they can be run in nearly all circumstances one would run the quantum algorithm.
- Solution (Tang): compare quantum algorithms with quantum state preparation to classical algorithms with sample and query access to input.

Classical ℓ^2 Sampling Model

Definition

We have ”query access” to $x \in \mathbb{C}^n$ if, given $i \in [n]$, we can efficiently compute x_i. We say that $x \in \mathcal{Q}$.

Definition

We have sample **and** query access to $x \in \mathbb{C}^n$ if

1. We have query access to x i.e. $x \in \mathcal{Q}$ ($\Rightarrow \mathcal{SQ} \subset \mathcal{Q}$)
2. can produce independent random samples $i \in [n]$ where we sample i with probability $|x_i|^2/\|x\|^2$ and can query for $\|x\|$.

We say that $x \in \mathcal{SQ}$.

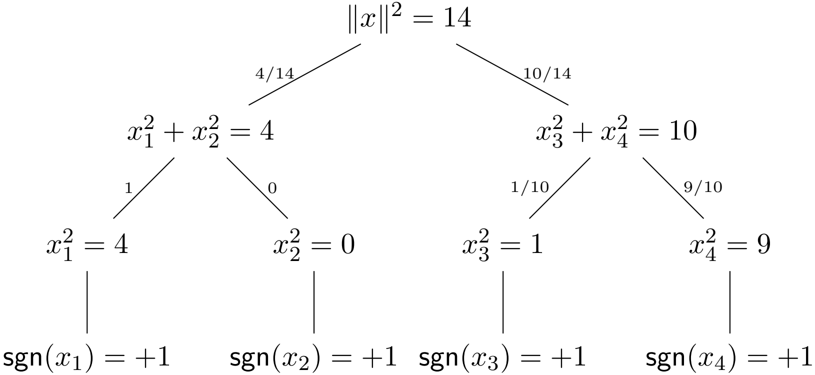
Definition

For $A \in \mathbb{C}^{m \times n}$, $A \in \mathcal{SQ}$ (abuse) if

1. $A_i \in \mathcal{SQ}$ where A_i is the ith row of A
2. $\tilde{A} \in \mathcal{SQ}$ for $\tilde{A}$ the vector of row norms (so $\tilde{A}_i = \|A_i\|$).

Example Data Structure

Say we have the vector $\vec{x} = (2, 0, 1, 3)$ and $\vec{x} \in \mathcal{SQ}$. Consider the following binary tree data structure.



$\|x\|^2 = 14$

$4/14$ $10/14$

$x_1^2 + x_2^2 = 4$ $x_3^2 + x_4^2 = 10$

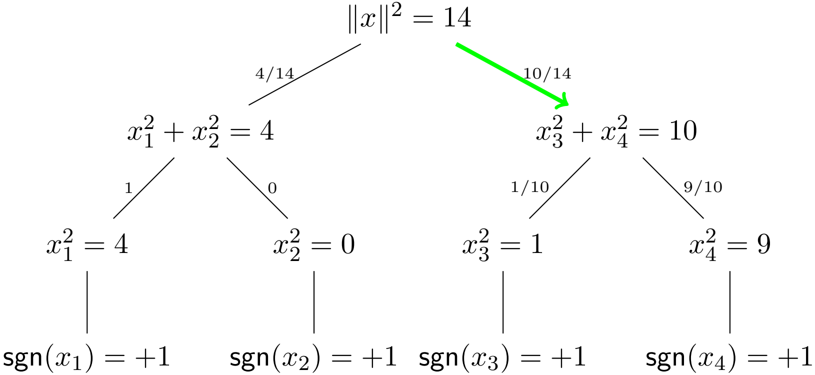
1 0 $1/10$ $9/10$

$x_1^2 = 4$ $x_2^2 = 0$ $x_3^2 = 1$ $x_4^2 = 9$

$\mathsf{sgn}(x_1) = +1$ $\mathsf{sgn}(x_2) = +1$ $\mathsf{sgn}(x_3) = +1$ $\mathsf{sgn}(x_4) = +1$

Example Data Structure

Say we have the vector $\vec{x} = (2, 0, 1, 3)$ and $\vec{x} \in \mathcal{SQ}$. Consider the following binary tree data structure.



$\|x\|^2 = 14$

$4/14$

$10/14$

$x_1^2 + x_2^2 = 4$

$x_3^2 + x_4^2 = 10$

1

0

$1/10$

$9/10$

$x_1^2 = 4$

$x_2^2 = 0$

$x_3^2 = 1$

$x_4^2 = 9$

$\text{sgn}(x_1) = +1$

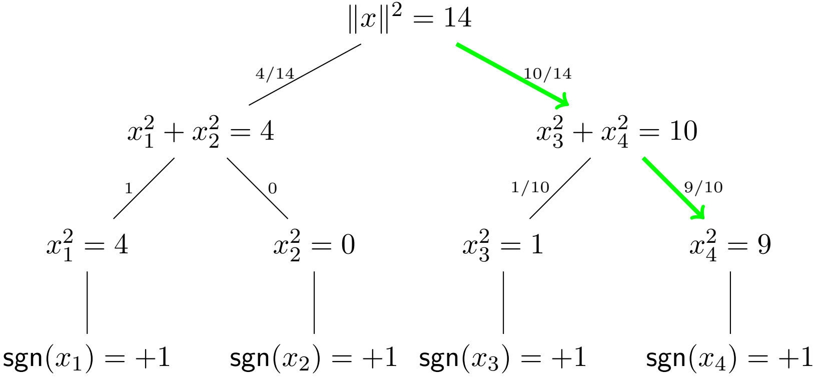
$\text{sgn}(x_2) = +1$

$\text{sgn}(x_3) = +1$

$\text{sgn}(x_4) = +1$

Example Data Structure

Say we have the vector $\vec{x} = (2, 0, 1, 3)$ and $\vec{x} \in \mathcal{SQ}$. Consider the following binary tree data structure.



Dequantization Toolbox

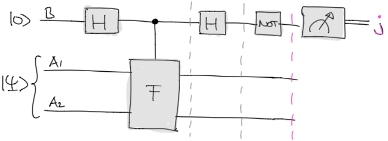
Method 1: Inner product estimation (Tang, 2018)

- For $x, y \in \mathbb{C}^n$, if we are given that $x \in \mathcal{SQ}$ and $y \in \mathcal{Q}$, then we can estimate $\langle x, y \rangle$ with probability $\geq 1 - \delta$ and error $\epsilon \|x\| \|y\|$

Dequantization Toolbox

Method 1: Inner product estimation (Tang, 2018)

- For $x, y \in \mathbb{C}^n$, if we are given that $x \in \mathcal{SQ}$ and $y \in \mathcal{Q}$, then we can estimate $\langle x, y \rangle$ with probability $\geq 1 - \delta$ and error $\epsilon \|x\| \|y\|$
- Quantum analog: SWAP test



Dequantization Toolbox

Method 1: Inner product estimation (Tang, 2018)

Fact

For $\{X_{i,j}\}$ i.i.d random variables with mean μ and variance σ^2, let

$$Y := \operatorname*{median}_{j \in [\log 1/\delta]} \operatorname*{mean}_{i \in [1/\epsilon^2]} X_{i,j}$$

Then $|Y - \mu| \leq \epsilon\sigma$ with probability $\geq 1 - \delta$, using only $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ samples.

- In words: We may create a mean estimator from $1/\epsilon^2$ samples of X. We compute the median of $\log 1/\delta$ such estimators

Dequantization Toolbox

Method 1: Inner product estimation (Tang, 2018)

Fact

For $\{X_{i,j}\}$ i.i.d random variables with mean μ and variance σ^2, let

$$Y := \underset{j \in [\log 1/\delta]}{\text{median}} \ \underset{i \in [1/\epsilon^2]}{\text{mean}} \ X_{i,j}$$

Then $|Y - \mu| \leq \epsilon\sigma$ with probability $\geq 1 - \delta$, using only $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ samples.

- In words: We may create a mean estimator from $1/\epsilon^2$ samples of X. We compute the median of $\log 1/\delta$ such estimators
- Catoni (2012) shows that Chebyshev's inequality is the best guarantee one can provide when considering pure empirical mean estimators for an unknown distribution (and finite μ, σ)
- "Median of means" provides an exponential improvement in probability of success $(1 - \delta)$ guarantee

Dequantization Toolbox

Method 1: Inner product estimation (Tang, 2018)

Corollary

For $x, y \in \mathbb{C}^n$, given $x \in \mathcal{S}\mathcal{Q}$ and $y \in \mathcal{Q}$, we can estimate $\langle x, y \rangle$ to $\epsilon\|x\|\|y\|$ error with probability $\geq 1 - \delta$ with query complexity $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$

Dequantization Toolbox

Method 1: Inner product estimation (Tang, 2018)

Corollary

For $x, y \in \mathbb{C}^n$, given $x \in \mathcal{SQ}$ and $y \in \mathcal{Q}$, we can estimate $\langle x, y \rangle$ to $\epsilon \|x\| \|y\|$ error with probability $\geq 1 - \delta$ with query complexity $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$

Proof.

Sample an **index** s from x. Then, define $Z := x_s y_s \frac{\|y\|^2}{|y_s|^2}$. Apply the Fact with $X_{i,j}$ being independent samples Z. □

Dequantization Toolbox

Method 2: Thin Matrix-Vector (Tang, 2018)

- For $V \in \mathbb{C}^{n \times k}, w \in \mathbb{C}^{k}$, given $V^{\dagger} \in \mathcal{SQ}$ (*column*-wise sampling of V) and $w \in \mathcal{Q}$, we can simulate $Vw \in \mathcal{SQ}$ with poly(k) queries
- In words: if we can least-square sample the columns of matrix V and query the entries of vector w, then
  1. We can query entries of their multiplication (Vw)
  2. We can least-square sample from a distribution that emulates their multiplication
- Hence, as long as $k \ll n$, we can perform each using a number of steps polynomial in the number of columns of V.

Dequantization Toolbox

Method 2: Thin Matrix-Vector (Tang, 2018)

Definition

Rejection sampling

Algorithm

Input: Samples from distribution P

Output: Samples from distribution Q

- *Sample s from P*
- *Compute $r_s = \frac{1}{N}\frac{Q(s)}{P(s)}$, for fixed constant N*
- *Output s with probability r_s and restart otherwise*

Fact

Fact. If $r_i \leq 1, \forall i$, then the above procedure is well-defined and outputs a sample from Q in N iterations in expectation.

Dequantization Toolbox

Method 2: Thin Matrix-Vector (Tang, 2018)

Proposition

For $V \in \mathbb{R}^{n \times k}$ and $w \in \mathbb{R}^k$, given $V^\dagger \in \mathcal{SQ}$ and $w \in \mathcal{Q}$, we can simulate $Vw \in \mathcal{SQ}$ with expected query complexity $\tilde{O}((\frac{1}{\epsilon^2} \log \frac{1}{\delta}))$
We can compute entries $(Vw)_i$ with $O(k)$ queries.
We can sample using rejection sampling:

- *P is the distribution formed by sampling from $V_{(\cdot,j)}$.*
- *Q is the target Vw.*
- *Hence, compute r_s to be a constant factor of Q/P*

$$r_i = \frac{\|w^T V_{\cdot,i}\|^2}{\|w\|^2 \|V_{\cdot,i}\|^2}$$

Dequantization Toolbox

Method 2: Thin Matrix-Vector (Tang, 2018)

- Notice that we can compute these r_i's (in fact, despite that we cannot compute probabilities from the target distribution), and that the rejection sampling guarantee is satisfied (via Cauchy-Schwarz).
- Since the probability of success is $\|Vw\|^2/\|w\|^2$, it suffices to estimate the probability of success of this rejection sampling process to estimate this norm.
- Through a Chernoff bound, we see that the average of $O(\|w\|^2(\frac{1}{\epsilon^2}\log\frac{1}{\delta}))$ "coin flips" is in $[(1-\epsilon)\|Vw\|, (1+\epsilon)\|Vw\|]$ with probability $\geq 1-\delta$.

Dequantization Toolbox

Method 3: Low-Rank Approximation (Frieze, Kannan, Vempala, 1998)

- For $A \in \mathbb{C}^{m\times n}$, given $A \in \mathcal{SQ}$ and some threshold k, we can output a description of a low-rank approximation of A with poly(k) queries.
- Specifically, we output two matrices $S, \hat{U} \in \mathcal{SQ}$ where $S \in \mathbb{C}^{\ell\times n}$, $\hat{U} \in \mathbb{C}^{\ell\times k}$ ($\ell =$ poly$(k, \frac{1}{\epsilon})$), and this implicitly describes the low-rank approximation to A, $D := A(S^\dagger \hat{U})(S^\dagger \hat{U})^\dagger$ ($\Rightarrow$ rank $D \leq k$).
- This matrix satisfies the following low-rank guarantee with probability $\geq 1-\delta$: for $\sigma := \sqrt{2/k}\|A\|_F$, and $A_\sigma := \sum_{\sigma_i \geq \sigma} \sigma_i u_i v_i^\dagger$ (using SVD),

$$\|A - D\|_F^2 \leq \|A - A_\sigma\|_F^2 + \epsilon^2 \|A\|_F^2$$

- Note the $\|A - A_\sigma\|_F^2$ term. This says that our guarantee is weak if A has no large singular values.
- Quantum analog: phase estimation

Dequantization Toolbox

$$\left[\cdots A \cdots\right] \left[S^\dagger\right] [\hat{U}] \left[\hat{U}^\dagger\right] [\cdots S \cdots]$$

Moore-Penrose Pseudoinverse (low-rank)

Application (Lloyd, Tang, 2018)

Problem

For a low-rank matrix $A \in \mathbb{R}^{m \times n}$ *and a vector* $b \in \mathbb{R}^n$*, given* $b, A \in \mathcal{SQ}$*, (approximately) simulate* $A^+ b \in \mathcal{SQ}$*.*

Moore-Penrose Pseudoinverse (low-rank)

Application (Lloyd, Tang, 2018)

Problem

For a low-rank matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^n$, given $b, A \in \mathcal{SQ}$, (approximately) simulate $A^+ b \in \mathcal{SQ}$.

Algorithm

- *Low-rank approximation (3) gives us $S, \hat{U} \in \mathcal{SQ}$.*
- *Applying thin-matrix vector (2), we get $\hat{V} \in \mathcal{SQ}$, where $\hat{V} := S^T \hat{U}$; we can show that the columns of $\hat{V}$ behave like the right singular vectors of A.*
- *Let $\hat{U}$ have columns $\{\hat{u}_i\}$. Hence, $\hat{V}$ has columns $\{S\hat{u}_i\}$. Write its ith column as $\hat{v}_i := S\hat{u}_i$.*
- *Low-rank approximation (3) also outputs the approximate singular values $\hat{\sigma}_i$ of A*

Moore-Penrose Pseudoinverse (low-rank) cont.

Application (Lloyd, Tang, 2018)

Now, we can write the approximate vector we wish to sample in terms of these approximations:

$$A^+ b = (A^T A)^+ A^T b \approx \sum_{i=1}^{k} \frac{1}{\hat{\sigma}_i^2} \hat{v}_i \hat{v}_i^T A^T b$$

Moore-Penrose Pseudoinverse (low-rank) cont.

Application (Lloyd, Tang, 2018)

- We approximate $\hat{v}_i^T A^T b$ to additive error for all by noticing that $\hat{v}_i^T A^T b = \operatorname{tr}\left(A^T b \hat{v}_i^T\right)$ is an inner product of A^T and $b\hat{v}_i^T$.
- Thus, we can apply (1), since being given $A \in \mathcal{SQ}$ implies $A^T \in \mathcal{SQ}$ for A^T viewed as a long vector.
- Define the approximation of $\hat{v}_i^T A^T b$ to be $\hat{\lambda}_i$. At this point we have (recalling that $\hat{v}_i := S\hat{u}_i$)

$$A^+ b \approx \sum_{i=1}^{k} \frac{1}{\hat{\sigma}_i^2} \hat{v}_i \hat{\lambda}_i = S \sum_{i=1}^{k} \frac{1}{\hat{\sigma}_i^2} \hat{u}_i \hat{\lambda}_i$$

- Finally, using (2) to provide sample access to each $S\hat{u}_i$, we are done ! $\tilde{O}(\kappa^{16} k^6 \|A\|_F^6 / \epsilon^6)$ complexity.

Machine Learning

Quantum Machine Learning

Classical ℓ^2 sampling

Remarks

Thoughts

- Claim (Tang): For machine learning problems, $\mathcal{SQ}$ assumptions are more reasonable than state preparation assumptions.
- We discussed pseudo-inverse which inverts singular values, but in principle we could have applied any function to the singular values
- Gilyen et. al (2018) show that many quantum machine learning algorithms indeed apply polynomial functions to singular values
- Our discussion suggests that exponential quantum speedups are tightly related to problems where high-rank matrices play a crucial role (e.g. Hamiltonian simulation or QFT)

Thank you for listening!

Questions? fms15@duke.edu

Read the Fine Print

- This poses two problems if seek to use these algorithms: the "state preparation" and "readout" problems.
- Even if we ignore the readout problem, can we at least find a state preparation routine that maintains a speedup for the discussed quantum algorithms? Open question!
- See "Quantum Machine Learning Algorithms: Read the Fine Print" by Aaronson

"Dequantization" (Tang)

Definition

Let $\mathcal{A}$ be a quantum algorithm with input $|\varphi_1\rangle, \ldots, |\varphi_C\rangle$ and output either a state $|\psi\rangle$ or a value λ. We say we dequantize $\mathcal{A}$ if we describe a classical algorithm that, given $\varphi_1, \ldots, \varphi_C \in \mathcal{SQ}$, can evaluate queries to $\psi \in \mathcal{SQ}$ or output λ, with similar guarantees to $\mathcal{A}$ and query complexity poly(C).