Quantum Algorithms and Machine Learning

Iordanis Kerenidis
CNRS, IRIF, Univ Paris Diderot

Joint work with Anupam Prakash
Quantum Algorithms: the History I

Problem

How can we simulate the behaviour of quantum systems, since we need to keep track of an exponential number of parameters?
Quantum Algorithms: the History I

Problem

How can we simulate the behaviour of quantum systems, since we need to keep track of an exponential number of parameters?

Feynman’s Idea

- Use quantum systems to simulate quantum systems.
Quantum Algorithms: the History I

Problem
How can we simulate the behaviour of quantum systems, since we need to keep track of an exponential number of parameters?

Feynman’s Idea
- Use quantum systems to simulate quantum systems.

DWave
A system of 1024 quantum bits that can solve optimization problems (simulated annealing)
Quantum Algorithms: the History II

Factoring [Shor 94]  There exists a polynomial time quantum algorithms to factor large numbers
Quantum Algorithms: the History II

Factoring [Shor 94] There exists a polynomial time quantum algorithms to factor large numbers

Algorithm: Reduce factoring \( N \) to finding the period of modular exponentiation \( f(a) = x^a \mod N \)
Quantum Fourier Transform finds the period of a function efficiently.
Quantum Algorithms: the History II

Factoring [Shor 94] There exists a polynomial time quantum algorithms to factor large numbers

Algorithm: Reduce factoring $N$ to finding the period of modular exponentiation $f(a) = x^a (N)$

Quantum Fourier Transform finds the period of a function efficiently.

Good news: Quantum computers might be more powerful than classical computers!
Quantum Algorithms: the History II

Factoring [Shor 94] There exists a polynomial time quantum algorithms to factor large numbers

Algorithm: Reduce factoring $N$ to finding the period of modular exponentiation $f(a) = x^a (N)$
Quantum Fourier Transform finds the period of a function efficiently.

Good news: Quantum computers might be more powerful than classical computers!

Bad news: An adversary with a quantum computer can break most currently used security systems
Quantum Algorithms: the History II

Factoring [Shor 94] There exists a polynomial time quantum algorithms to factor large numbers

**Algorithm:**
Reduce factoring $N$ to finding the period of modular exponentiation $f(a) = x^a (N)$
Quantum Fourier Transform finds the period of a function efficiently.

**Good news:** Quantum computers might be more powerful than classical computers!

**Bad news:** An adversary with a quantum computer can break most currently used security systems

“will initiate a transition to quantum resistant algorithms in the not too distant future…. Our ultimate goal is to provide cost effective security against a potential quantum computer.”
Quantum Algorithms: the History III

Search [Grover 97] There exists a quantum algorithm to Search an N-item list in $O(\sqrt{N})$ queries
Quantum Algorithms: the History III

Search [Grover 97] There exists a quantum algorithm to Search an N-item list in $O(\sqrt{N})$ queries

Algorithm: Amplitude Amplification. Every iteration moves $O\left(\frac{1}{\sqrt{N}}\right)$ amplitude
There exists a quantum algorithm to search an N-item list in $O(\sqrt{N})$ queries.

**Algorithm:** Amplitude Amplification. Every iteration moves $O\left(\frac{1}{\sqrt{N}}\right)$ amplitude.

**Good news:** A large number of applications in query complexity: Find Collisions, Approximate Counting, Shortest Path, ...
Search [Grover 97]  There exists a quantum algorithm to Search an N-item list in $O(\sqrt{N})$ queries

Algorithm:  Amplitude Amplification. Every iteration moves $O\left(\frac{1}{\sqrt{N}}\right)$ amplitude

Good news:  A large number of applications in query complexity: Find Collisions, Approximate Counting, Shortest Path, ...

Bad news:  Only quadratic improvement. Almost impossible to see it in practice!
Quantum Algorithms: the History III

Search [Grover 97] There exists a quantum algorithm to Search an N-item list in $O(\sqrt{N})$ queries

Algorithm: Amplitude Amplification. Every iteration moves $O\left(\frac{1}{\sqrt{N}}\right)$ amplitude

Good news: A large number of applications in query complexity: Find Collisions, Approximate Counting, Shortest Path, ...

Bad news: Only quadratic improvement. Almost impossible to see it in practice!

Other algorithms Quantum walks, Communication Complexity, Streaming Algorithms, PAC Learning, ...
Quantum Algorithms: the HHL Algorithm

Problem
Given a matrix $A$, and a vector $b$, solve the corresponding system of linear equations, i.e. find $x$ such that $Ax=b$
Quantum Algorithms: the HHL Algorithm

Problem

Given a matrix $A$, and a vector $b$, solve the corresponding system of linear equations, i.e. find $x$ such that $Ax=b$

Applications

Everywhere!
Quantum Algorithms: the HHL Algorithm

Problem
Given a matrix A, and a vector b, solve the corresponding system of linear equations, i.e. find x such that Ax=b

Applications
Everywhere!

Classical Algorithms
in time polynomial in the dimension
Quantum Algorithms: the HHL Algorithm

Problem

Given a matrix $A$, and a vector $b$, solve the corresponding system of linear equations, i.e. find $x$ such that $Ax=b$

Applications

Everywhere!

Classical Algorithms

in time polynomial in the dimension

The quantum HHL algorithm [Harrow, Hassidim, Lloyd 2009]

Given access to a matrix $A$, and a vector $b$, there exists a quantum algorithm that runs in time poly-logarithmic in the dimension and solves the system $Ax=b$
HHL: the input conditions

1. The vector $b$, should be easily loaded in quantum memory

$$(b_1, b_2, \ldots, b_n) \rightarrow \sum_{i=1}^{n} b_i |i\rangle$$
HHL: the input conditions

1. The vector $b$, should be easily loaded in quantum memory

$$ (b_1, b_2, ..., b_n) \rightarrow \sum_{i=1}^{n} b_i |i\rangle $$

- OK when $b$ is close to uniform vector
HHL: the input conditions

1. The vector $b$, should be easily loaded in quantum memory

$$(b_1, b_2, \ldots, b_n) \rightarrow \sum_{i=1}^{n} b_i |i\rangle$$

- OK when $b$ is close to uniform vector
- Augment classical memory to store not only the $b_i$’s, but also partial sums of $b_i$’s. Then, $b$ can be loaded in time $O(\log n)$, by performing conditional rotations.

Extra preprocessing in linear time.
HHL: the input conditions

1. The vector $b$, should be easily loaded in quantum memory

$$(b_1, b_2, \ldots, b_n) \rightarrow \sum_{i=1}^{n} b_i |i\rangle$$

- OK when $b$ is close to uniform vector
- Augment classical memory to store not only the $b_i$’s, but also partial sums of $b_i$’s.
  Then, $b$ can be loaded in time $O(\log n)$, by performing conditional rotations.
  Extra preprocessing in linear time.

2. The matrix $A$ should be sparse and well-conditioned

- The running time of the quantum algorithm is polynomial in the sparsity $s$ per row of $A$, and in the condition number ($\kappa = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}}$)
HHL: the output

Problem

Given a matrix $A$, and a vector $b$, solve the corresponding system of linear equations, i.e. find $x$ such that $Ax=b$
HHL: the output

Problem

Given a matrix $A$, and a vector $b$, solve the corresponding system of linear equations, i.e. find $x$ such that $Ax=b$

The HHL output

Let $x$, the solution to the system $Ax=b$. Then, HHL outputs a quantum state that corresponds to the solution

$$ |x\rangle = \sum_{i=1}^{n} x_i |i\rangle $$
HHL: the output

Problem

Given a matrix A, and a vector b, solve the corresponding system of linear equations, i.e. find x such that Ax=b

The HHL output

Let x, the solution to the system Ax=b. Then, HHL outputs a quantum state that corresponds to the solution

\[ |x\rangle = \sum_{i=1}^{n} x_i |i\rangle \]

Remarks

- The output encodes the entries of x in its amplitudes
- It has \(\log n\) qubits, hence it has only \(\log n\) bits of information about x
- It can not be used to find a specific \(x_i\)
- It can be used: Find big coefficient; Estimate Inner Products
The HHL algorithm (for real!)

Problem

Given a matrix $A$, and a vector $b$, solve the corresponding system of linear equations, i.e. find $x$ such that $Ax=b$

The HHL algorithm [Harrow, Hassidim, Lloyd 2009]

Given access to a matrix $A$, and a vector $b$, such that

i. $b$ can be efficiently loaded in quantum memory;

ii. $A$ has sparsity $s$ per row and condition number $\kappa$,

there exists a quantum algorithm that outputs $|x\rangle$, where $Ax=b$, and runs in time polynomial in $(s, \kappa)$ and poly-logarithmic in the dimension
The HHL algorithm (for real!)

Problem
Given a matrix A, and a vector b, solve the corresponding system of linear equations, i.e. find x such that Ax=b

The HHL algorithm [Harrow, Hassidim, Lloyd 2009]
Given access to a matrix A, and a vector b, such that
i. b can be efficiently loaded in quantum memory;
ii. A has sparsity s per row and condition number \( \kappa \), there exists a quantum algorithm that outputs \(|x\rangle\), where Ax=b, and runs in time polynomial in (s, \( \kappa \)) and poly-logarithmic in the dimension

Remark
“Solving” systems of linear equations is BQP-complete
The HHL algorithm (for real!)

Problem

Given a matrix $A$, and a vector $b$, solve the corresponding system of linear equations, i.e. find $x$ such that $Ax=b$

The HHL algorithm [Harrow, Hassidim, Lloyd 2009]

Given access to a matrix $A$, and a vector $b$, such that

i. $b$ can be efficiently loaded in quantum memory;
ii. $A$ has sparsity $s$ per row and condition number $\kappa$,

there exists a quantum algorithm that outputs $|x\rangle$, where $Ax=b$, and runs in time polynomial in $(s, \kappa)$ and poly-logarithmic in the dimension

Remark

“Solving” systems of linear equations is BQP-complete

Main Question

What is it good for?
The HHL algorithm and Machine Learning

HHL initiated the field of Quantum Machine Learning

“Recent strides in quantum computing have raised the prospects that near term quantum devices can expediently solve computationally intractable problems in simulation, optimization and machine learning. The opportunities that quantum computing raises for machine learning is hard to understate. It opens the possibility of dramatic speedups for machine learning tasks, richer models for data sets and more natural settings for learning and inference than classical computing affords.”

Quantum Machine Learning Workshop during NIPS 2015
Applications: Data Fitting

Least Square Fitting [Wiebe, Braun, Lloyd 12]

Input: N labelled points \((x_i, y_i)\).

Output: A fit function \(f(x, \lambda) = \sum_{j=1}^{M} f_j(x) \lambda_j\) that minimizes the error

\[
E = \sum_{i=1}^{N} |f(x_i, \lambda) - y_i|^2 = |F\lambda - y|^2
\]

Note that \(\lambda = (F^*F)^{-1}F^*y\)
Applications: Data Fitting

Least Square Fitting [Wiebe, Braun, Lloyd 12]

**Input:** N labelled points \((x_i, y_i)\).

**Output:** A fit function \(f(x, \lambda) = \sum_{j=1}^{M} f_j(x) \lambda_j\) that minimizes the error

\[ E = \sum_{i=1}^{N} |f(x_i, \lambda) - y_i|^2 = |F\lambda - y|^2 \]

Note that \(\lambda = (F^\dagger F)^{-1}F^\dagger y\)

**Quantum Algorithm**

By using HHL, given quantum access to a matrix A and a vector b, we can apply A to the vector b, i.e. create the state \(|Ab\rangle=\sum_{i=1}^{N} A_i b |i\rangle\) and

Apply \(A^{-1}\) to the vector b, i.e. create the state \(|A^{-1}b\rangle=\sum_{i=1}^{N} (A^{-1})_i b |i\rangle\)

**Quantum output:** \(|\lambda\rangle\)
Applications: Support Vector Machines

Input: M labelled N-dimensional points \((x_j, y_j), x_j \in \mathbb{R}^N, y_j \in \{-1, 1\}\)

Output: A maximum margin hyper-plane that separates the classes
Applications: Support Vector Machines

Input: M labelled N-dimensional points \((x_j, y_j), x_j \in \mathbb{R}^N, y_j \in \{-1, 1\}\)

Output: A maximum margin hyper-plane that separates the classes

Remark: SVM can be recast as a system of linear equations
Applications: Support Vector Machines

Input: M labelled N-dimensional points \((x_j, y_j), x_j \in \mathbb{R}^N, y_j \in \{-1, 1\}\)

Output: A maximum margin hyper-plane that separates the classes

Remark: SVM can be recast as a system of linear equations

Quantum Algorithm

Use HHL to output the quantum state that corresponds to the normal vector of the hyperplane \(|w\rangle\).
Applications: Support Vector Machines

Input: M labelled N-dimensional points \((x_j, y_j), x_j \in \mathcal{R}^N, y_j \in \{-1, 1\}\)
Output: A maximum margin hyper-plane that separates the classes

Remark: SVM can be recast as a system of linear equations

Quantum Algorithm
Use HHL to output the quantum state that corresponds to the normal vector of the hyperplane \(|w\rangle\).

Application
Classify data, by estimating the inner product with \(w\).
Applications: Singular Value Estimation

Sampling eigen-values/vectors [Lloyd, Mohseni, Rebentrost 13]

Let $A$ a PSD, trace 1 matrix. There exists a quantum algorithm that efficiently samples an eigenvector with the corresponding eigenvalue.
Applications: Singular Value Estimation

Sampling eigen-values/vectors [Lloyd, Mohseni, Rebentrost 13]

Let A a PSD, trace 1 matrix. There exists a quantum algorithm that efficiently samples an eigenvector with the corresponding eigenvalue.

Singular Value Estimation [LMR 13, Prakash 15]

Let M with SVD $M = \sum_i \sigma_i u_i(v_i)^t$. There exists a quantum algorithm running in time $\text{polylog}(mn)/\epsilon$, that:

Given a singular vector $|v_i\rangle$,

outputs an estimate of the singular value $\bar{\sigma}_i \in \sigma_i \pm \epsilon \|M\|$. 
Applications: Singular Value Estimation

Sampling eigen-values/vectors [Lloyd, Mohseni, Rebentrost 13]

Let A a PSD, trace 1 matrix. There exists a quantum algorithm that efficiently samples an eigenvector with the corresponding eigenvalue.

Singular Value Estimation [LMR 13, Prakash 15]

Let M with SVD $M = \sum \sigma_i u_i(v_i)^t$. There exists a quantum algorithm running in time $\text{polylog}(mn)/\varepsilon$, that:

Given a singular vector $|v_i\rangle$,
outputs an estimate of the singular value $\tilde{\sigma}_i \in \sigma_i \pm \varepsilon \|M\|$.

Singular Value Estimation (Coherent version)

Let M with SVD $M = \sum \sigma_i u_i(v_i)^t$. There exists a quantum algorithm running in time $\text{polylog}(mn)/\varepsilon$, that maps:

$$\sum_i a_i |v_i\rangle \rightarrow \sum_i a_i |v_i\rangle |\tilde{\sigma}_i\rangle \text{ with } \tilde{\sigma}_i \in \sigma_i \pm \varepsilon \|M\|$$
A new Application in Machine Learning

Competitive Recommendation Systems

- tracks purchases of a group of users.
- makes product recommendations to individual users
A new Application in Machine Learning

Competitive Recommendation Systems

- tracks purchases of a group of users.
- makes product recommendations to individual users.

Netfliex Prize

What we were interested in:
- High quality recommendations

Proxy question:
- Accuracy in predicted rating
- Improve by 10% = $1million!

Results
- Top 2 algorithms still in production

RMSE = \sqrt{\frac{1}{n} \sum_{j=1}^{n} (y_j - \hat{y}_j)^2}
A new Application in Machine Learning

Competitive Recommendation Systems
- tracks purchases of a group of users.
- makes product recommendations to individual users

In this talk
- Model
- Matrix reconstruction problem and sampling.
- Quantum algorithm that runs in time polylogarithmic in the size of the matrix and polynomially in rank. (based on HHL)
## Model: Preference Matrix A

<table>
<thead>
<tr>
<th>User</th>
<th>Product 1</th>
<th>Product 2</th>
<th>...</th>
<th>Product n</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>0.1</td>
<td>0.6</td>
<td>...</td>
<td>0.2</td>
</tr>
<tr>
<td>User 2</td>
<td>0.3</td>
<td>0.2</td>
<td>...</td>
<td>0.8</td>
</tr>
<tr>
<td>User m</td>
<td>0.9</td>
<td>0.1</td>
<td>...</td>
<td>0.3</td>
</tr>
</tbody>
</table>

\( A_{ij} \): preference of user i for product j

**Assumption:** A has a good low-rank approximation
Model: Recommendation Matrix $T$

<table>
<thead>
<tr>
<th></th>
<th>Product 1</th>
<th>Product 2</th>
<th>...</th>
<th>Product n</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>User 2</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>User m</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

1: good recommendation, 0: bad recommendation

Assumption: $T$ has a good low-rank approximation
Matrix Reconstruction of matrix $T$

<table>
<thead>
<tr>
<th>User 1</th>
<th>Product 1</th>
<th>Product 2</th>
<th>...</th>
<th>Product n</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 2</td>
<td>?</td>
<td>1</td>
<td>...</td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>?</td>
<td>0</td>
<td>...</td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>User m</td>
<td>1</td>
<td>?</td>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

0/1: already known information about users/products
?: missing information

Goal: Given samples from $T$, find a matrix $T'$ close to $T$
Matrix Reconstruction: classical algorithms

Subsample Algorithm [Achlioptas, McSherry STOC 2001]

- **Input:** A matrix $\hat{T}$, where
  \[
  \hat{T}_{ij} = \begin{cases} 
  T_{ij}/p & \text{with prob. } p \\
  0 & \text{o/w}
  \end{cases}
  \]
- **Output:** $\hat{T}_k$, the projection on the $k$-top singular vectors
Matrix Reconstruction: classical algorithms

Subsample Algorithm [Achlioptas, McSherry STOC 2001]

- **Input:** A matrix $\hat{T}$, where
  \[
  \hat{T}_{ij} = \begin{cases} 
  T_{ij} / p & \text{with prob. } p \\
  0 & \text{o/w}
  \end{cases}
  \]
- **Output:** $\hat{T}_k$, the projection on the k-top singular vectors

**Theorem**
\[
\|T - \hat{T}_k\| \leq \|T - T_k\| + \|T_k - \hat{T}_k\| \leq \mu \|T\| + \epsilon \|T\|
\]
both for Frobenious and $2$-norm
Matrix Reconstruction: classical algorithms

CR-Algorithm [Drineas, K. Raghavan STOC 2002]

\[
\begin{pmatrix}
A \\
0 \\
0
\end{pmatrix}
\rightarrow
\begin{pmatrix}
\tilde{A}
\end{pmatrix}
\rightarrow
\begin{pmatrix}
C
\end{pmatrix}
\rightarrow
\tilde{U}_k
\rightarrow
\tilde{U}_k \tilde{U}_k^T \tilde{A}
\]

Theorem

\[
\|T - \hat{T}_k\| \leq \|T - T_k\| + \|T_k - \hat{T}_k\| \leq \mu \|T\| + \varepsilon \|T\| \quad \text{(Frobenious)}
\]
Matrix Reconstruction: Is it necessary?

- Good recommendations reduces to computing $\hat{T}_k$, a close approximation to $T$
Matrix Reconstruction: Is it necessary?

- Good recommendations reduces to computing $\hat{T}_k$, a close approximation to $T$.

- It suffices to get the high elements of $\hat{T}_k$. 
Matrix Reconstruction: Is it necessary?

- Good recommendations reduces to computing $\hat{T}_k$, a close approximation to $T$.

- It suffices to get the high elements of $\hat{T}_k$.

- $T$ is a 0-1 matrix. It suffices to sample from $\hat{T}_k$!
Recommendations via Sampling

Theorem 1

Let $\tilde{T}$ be an approximation of a matrix $T$, s.t. $\|T - \tilde{T}\|_F^2 \leq \varepsilon \|T\|_F^2$. Then, if we recommend a sample according to $\tilde{T}$, we have $\Pr[(i,j) \text{ is bad rec.}] \leq \varepsilon$. ($\tilde{T} = \hat{T}_k$)
Recommendations via Sampling

Theorem 1

Let $\tilde{T}$ be an approximation of a matrix $T$, s.t. $\|T - \tilde{T}\|_F^2 \leq \varepsilon\|T\|_F^2$.

Then, if we recommend a sample according to $\tilde{T}$, we have

$\Pr[(i,j) \text{ is bad rec.}] \leq \varepsilon$. ($\tilde{T} = \hat{T}_k$)

Remark 1

We need recommendations for specific users, hence we need to sample the row $(\hat{T}_k)_i$.
Recommendations via Sampling

Theorem 1

Let $\tilde{T}$ be an approximation of a matrix $T$, s.t. $\|T - \tilde{T}\|_F^2 \leq \epsilon \|T\|_F^2$. Then, if we recommend a sample according to $\tilde{T}$, we have $\Pr[(i,j) \text{ is bad rec.}] \leq \epsilon$. ($\tilde{T} = \hat{T}_k$)

Remark 1

We need recommendations for specific users, hence we need to sample the row $(\hat{T}_k)_i$

Theorem 2

For most users, for which there are enough good recommendations, a sample from $(\hat{T}_k)_i$ will be a good recommendation with high probability.
Recommendations via Sampling

Theorem 1

Let $\tilde{T}$ be an approximation of a matrix $T$, s.t. $\|T - \tilde{T}\|_F^2 \leq \varepsilon \|T\|_F^2$. Then, if we recommend a sample according to $\tilde{T}$, we have $\Pr[(i,j) \text{ is bad rec.}] \leq \varepsilon$. ($\tilde{T} = \hat{T}_k$)

Remark 1

We need recommendations for specific users, hence we need to sample the row $(\hat{T}_k)_i$.

Theorem 2

For most users, for which there are enough good recommendations, a sample from $(\hat{T}_k)_i$ will be a good recommendation with high probability.

Remark 2

$(\hat{T}_k)_i$ is the projection of $\hat{T}_i$ on the space spanned by the $k$-top column singular vectors of the matrix $\hat{T}$.
Recommendations via Quantum Sampling

Goal

Sample from \((\hat{T}_k)_i\), i.e. the projection of \(\hat{T}_i\) on the space spanned by the k-top singular vectors of the matrix \(\hat{T}\).
Recommendations via Quantum Sampling

Goal

Sample from \((\hat{T}_k)_i\), i.e. the projection of \(\hat{T}_i\) on the space spanned by the k-top singular vectors of the matrix \(\hat{T}\).

More general goal

Given a vector \(x\) and a matrix \(M\), sample from the projection of the vector on the space spanned by the column singular vectors of \(M\) with singular value higher than \(t\).
Recommendations via Quantum Sampling

Goal

Sample from \((\hat{T}_k)_i\), i.e. the projection of \(\hat{T}_i\) on the space spanned by the k-top singular vectors of the matrix \(\hat{T}\).

More general goal

Given a vector \(x\) and a matrix \(M\), sample from the projection of the vector on the space spanned by the column singular vectors of \(M\) with singular value higher than \(t\).

Quantum Solution

Given a vector \(x\) and a matrix \(M\), there exists a quantum algorithm that runs in time polylogarithmic in the dimension, that outputs the quantum state that corresponds to the projection of \(x\) onto the space spanned by the column singular vectors of \(M\) with singular value higher than \(t\).
Quantum Recommendation Systems

Input: a matrix $\hat{T}$ (norm=1), a user’s index $i$
Output: Product $j$
Quantum Recommendation Systems

Input: a matrix \( \hat{T} \) (norm=1), a user’s index \( i \)
Output: Product \( j \)

1. Apply quantum procedure to create the projection of the vector \( \hat{T}_i \), onto the space spanned by the singular vectors of \( \hat{T} \), with singular value at least \( \varepsilon / k \), and get the state \( |(\hat{T}_{ki})_i \rangle \)
Quantum Recommendation Systems

Input: a matrix $\hat{T}$ (norm=1), a user’s index $i$

Output: Product $j$

1. Apply quantum procedure to create the projection of the vector $\hat{T}_i$, onto the space spanned by the singular vectors of $\hat{T}$, with singular value at least $\varepsilon/k$, and get the state $|\hat{T}_{ki})_i\rangle$

2. Measure in the computational basis to sample $j$. 
Quantum Recommendation Systems

Input: a matrix $\hat{T}$ (norm=1), a user’s index $i$
Output: Product $j$

1. Apply quantum procedure to create the projection of the vector $\hat{T}_i$, onto the space spanned by the singular vectors of $\hat{T}$, with singular value at least $\varepsilon/k$, and get the state $|\{\hat{T}_{ki}\}_i\rangle$

2. Measure in the computational basis to sample $j$.

Running time: $O(k \times \text{polylog}(mn)/\varepsilon(1 - \varepsilon))$
Quantum Recommendation Systems

Input: a matrix $\hat{T}$ (norm=1), a user’s index $i$
Output: Product $j$

1. Apply quantum procedure to create the projection of the vector $\hat{T}_i$, onto the space spanned by the singular vectors of $\hat{T}$, with singular value at least $\epsilon/k$, and get the state $|\{\hat{T}_k\}_i\rangle$
2. Measure in the computational basis to sample $j$.

Running time: $O(k \cdot \text{polylog}(mn)/\epsilon(1 - \epsilon))$

Main Theorem

There exists a quantum algorithm that runs in time $\text{poly}(k)\text{polylog}(mn)$ and produces a sample from $(\hat{T}_k)_i$, such that for most users this sample will be a good recommendation with high probability.
Quantum Running Time

$m$ users, $n$ products

1. Preprocessing time
   - Time to gather the information of all users.
   - **Classically:** $n$ coefficients per user, $O(\text{sparsity})$
   - **Quantumly:** $2n$ coefficients (partial sums), $O(\text{sparsity})$
Quantum Running Time

$m$ users, $n$ products

1. Preprocessing time
   Time to gather the information of all users.
   Classically: $n$ coefficients per user, $O(\text{sparsity})$
   Quantumly: $2n$ coefficients (partial sums), $O(\text{sparsity})$

2. Online Recommendation time
   A user comes in the system. Time to recommend a product
   Classically: $O(n)$ (the algorithm reconstructs the row)
   Quantumly: $\text{polylog}(n)$ (the quantum algorithm samples row)
Quantum Projection

Input: a vector $x$, a matrix $M$ (norm=$1$, $M = \sum_i \sigma_i u_i (v_i)^T$), a threshold $t$

Output: the state $|M_t M_t^+ x\rangle$, where $M_t = \sum_i \sigma_i u_i (v_i)^T$, with $\sigma_i \geq t$
Quantum Projection

Input: a vector \( x \), a matrix \( M \) (norm=1, \( M = \sum_i \sigma_i u_i (v_i)^T \)), a threshold \( t \)

Output: the state \( |M_tM_t^+x\rangle \), where \( M_t = \sum_i \sigma_i u_i (v_i)^T \), with \( \sigma_i \geq t \)

1. Create \( |x\rangle = \frac{1}{|x|} \sum_{i=1}^n x_i |i\rangle \equiv |MM^+x\rangle \sum_i a_i |v_i\rangle + \sqrt{1 - |MM^+x|^2} \sum_i b_i |f_i\rangle \)

   Express is as a combination of singular vectors and orthogonal vectors

2. Do Singular Value Estimation

   \[ |MM^+x\rangle \sum_i a_i |v_i\rangle |\sigma_i\rangle + \sqrt{1 - |MM^+x|^2} \sum_i b_i |f_i\rangle |\sigma_i'\rangle \]

3. Map \( \sigma_i \geq t \) to 0 and get

   \[ |\varphi\rangle = |M_tM_t^+x\rangle |M_tM_t^+x\rangle |0\rangle + \sqrt{1 - |M_tM_t^+x|^2} |M_tM_t^+x\rangle^\perp |1\rangle \]

4. Measure second register until outcome \( |0\rangle \). Then, outcome \( |M_tM_t^+x\rangle \)
Conclusions and open questions

- Quantum recommendation systems “exponentially faster” than classical recommendation systems

- Quantum Machine Learning
  - Clarify the basics
  - Find real applications
  - Bring algorithms closer to implementation

- More quantum algorithms
  - Computational Learning / quantum-resistant cryptography