Quantum Machine Learning for Real-World, Large Scale Datasets with Applications in Earth Observation

Soronzonbold Otgonbaatar[1,2], Mihai Datcu[1], Xiao Xiang Zhu[3], and Dieter Kranzlmüller[2]
[1]German Aerospace Center (DLR) Oberpfaffenhofen, [2]Ludwig-Maximilians-Universität München (LMU München), [3]Technical University of Munich (TUM)

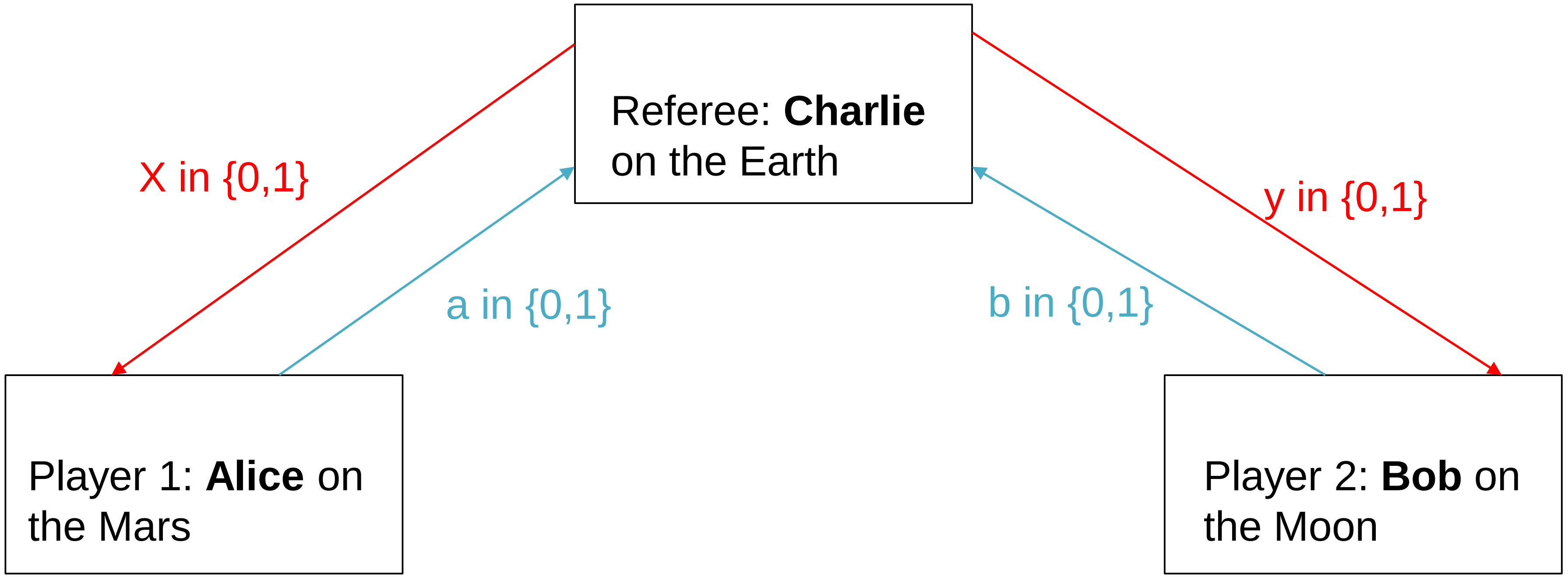
Contact: Soronzonbold.Otgonbaatar@dlr.de

AI4EO Symposium, Technical University of Munich (TUM)
13.10 – 14.10.2022

Contents

1. Quantum vs classical physics
2. Quantum vs classical computer
3. Machine learning on quantum computer or Quantum Machine Learning with **3** applications in Earth Observation

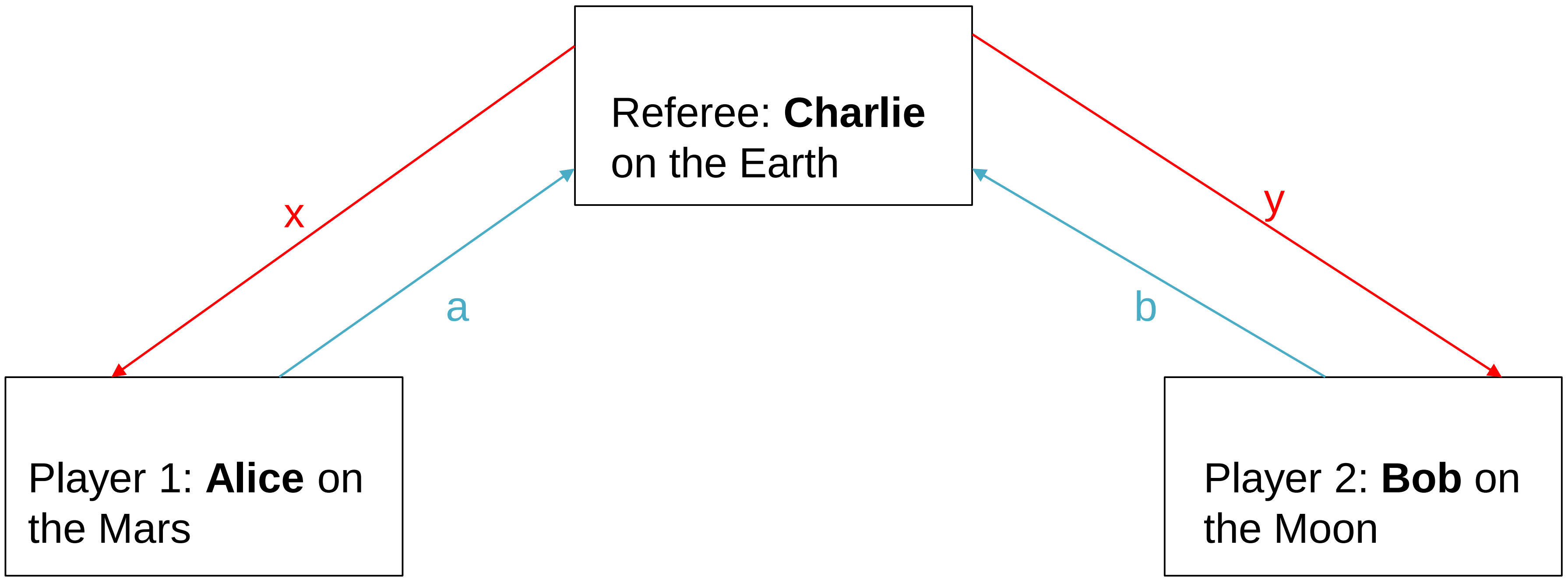
CHSH game



Players win If

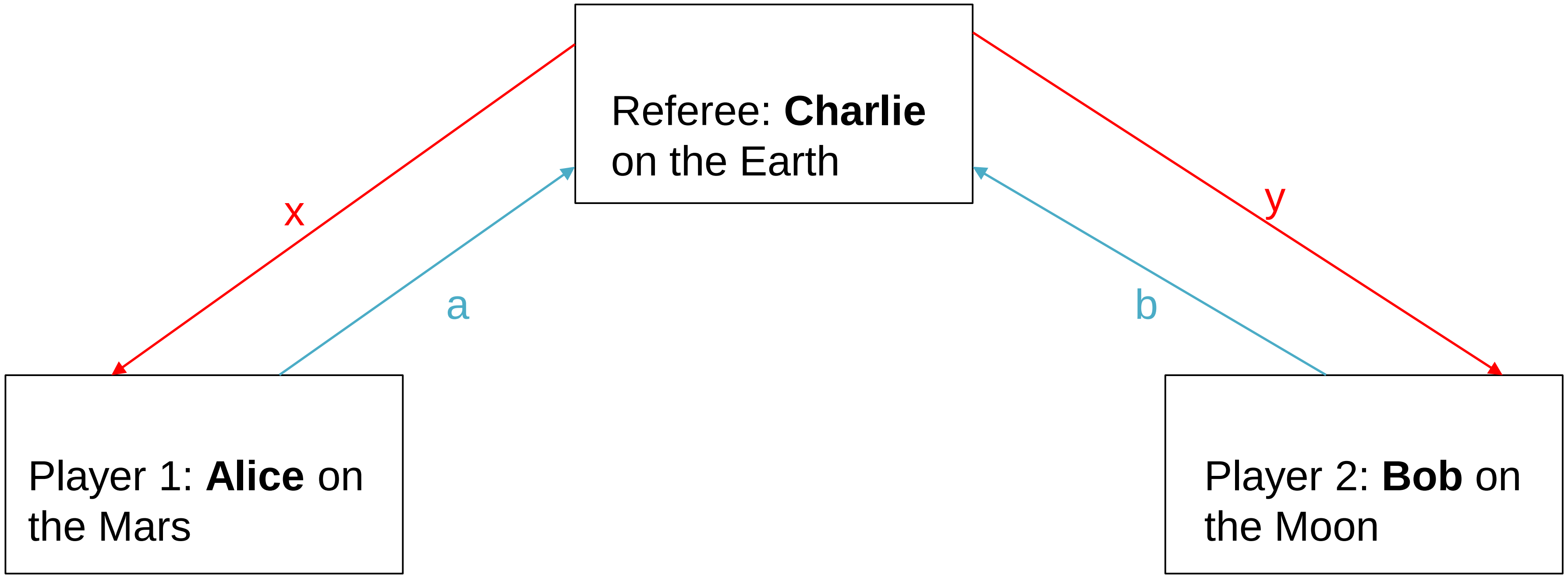
$$x \cdot y = a(xor)b;\ x, y, a, b \in \{0,1\}$$

CHSH game: classical world



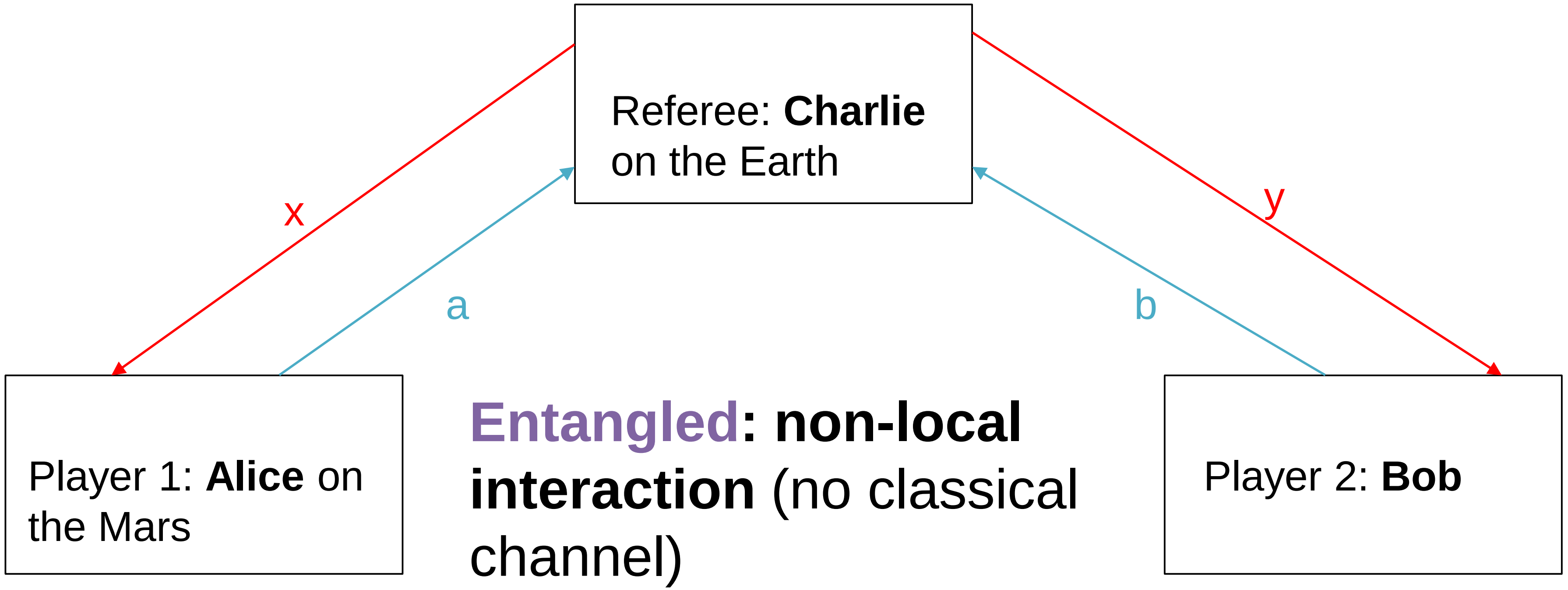
Players winning probability in **classical world: 75 percent**

CHSH game: quantum world



They share a so-called entangled quantum bits (particles).

CHSH game: quantum world



Entangled: non-local interaction (no classical channel)

Players winning probability in **quantum world: 85 percent**

Quantum Machine Learning

States entangled in a **quantum computer** yield **higher correlation values** (saw in CHSH game) than states in a **classical computer**. Classical Machine Learning involves the concepts of **probability and correlation**. Thus, this validates to study Machine Learning and deploy it on a **quantum computer**: Quantum Machine Learning (QML)

Classical & Quantum computer

0001110000001111111

CC: bits

|0001110000001111111> =
c1|110000110001111100>+
c2|010100100101100100>+
…
cn|010100110001110111>

QC: quantum bits (or qubits) which can exist in **superposition** and are **entangled**.

Classical & Quantum computer

Transistor error:

$$p \sim 10^{-27}$$

Qubit error:

$$p \sim 10^{-3}$$

CC: bits

QC: quantum bits (or qubits)

P. Shivakumar, M. Kistler, S. W. Keckler, D. Burger and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," *Proceedings International Conference on Dependable Systems and Networks*, 2002, pp. 389-398, doi: 10.1109/DSN.2002.1028924.

Classical & Quantum computer

Qubit error:

$$p \sim 10^{-3}$$

An error-corrected quantum computer, say $p \sim 10^{-27}$, is called a **fault-tolerant quantum computer**, and a **noisy-intermediate scale quantum computer (NISQ),** say $p \sim 10^{-13}$, otherwise.

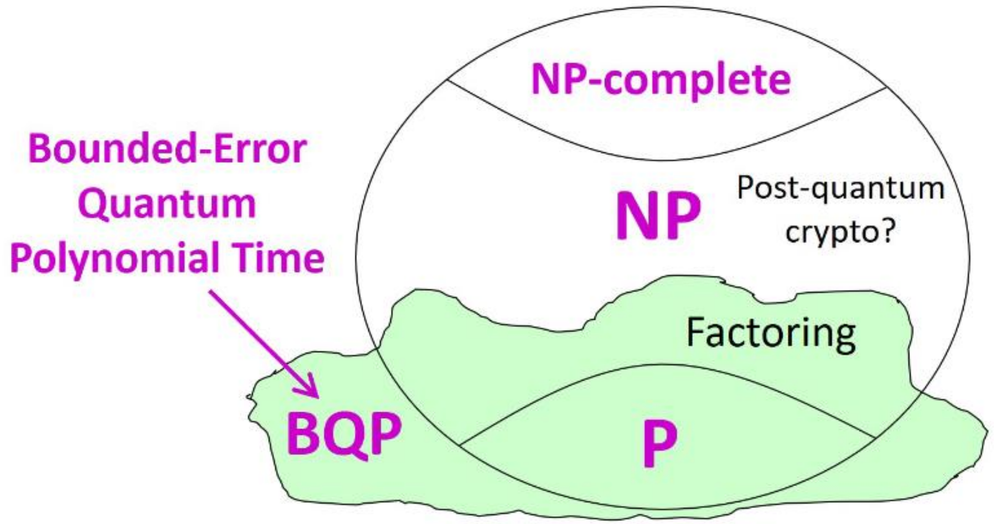
QC: quantum bits (or qubits)

John Preskill, Fault-tolerant quantum computer, arXiv: quant-ph/9712048
John Preskill, Quantum Computing in the NISQ era and beyond, arXiv: 1801.00862

Computational Complexity

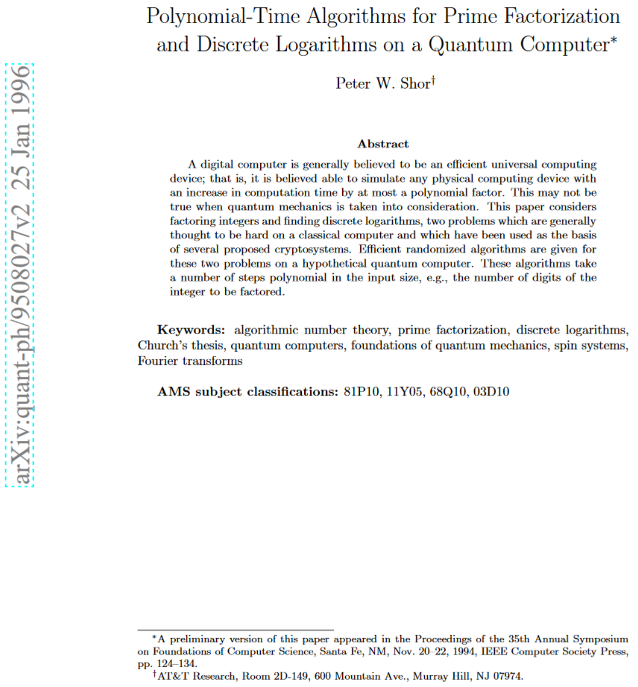
Classical Computer

Fault-Tolerant Quantum Computer



Scott Aaronson , How Much Structure Is Needed for Huge Quantum Speedups?, arXiv:2209.06930

Quantum Algorithm Evolution


arXiv:quant-ph/9508027v2 25 Jan 1996

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

Abstract

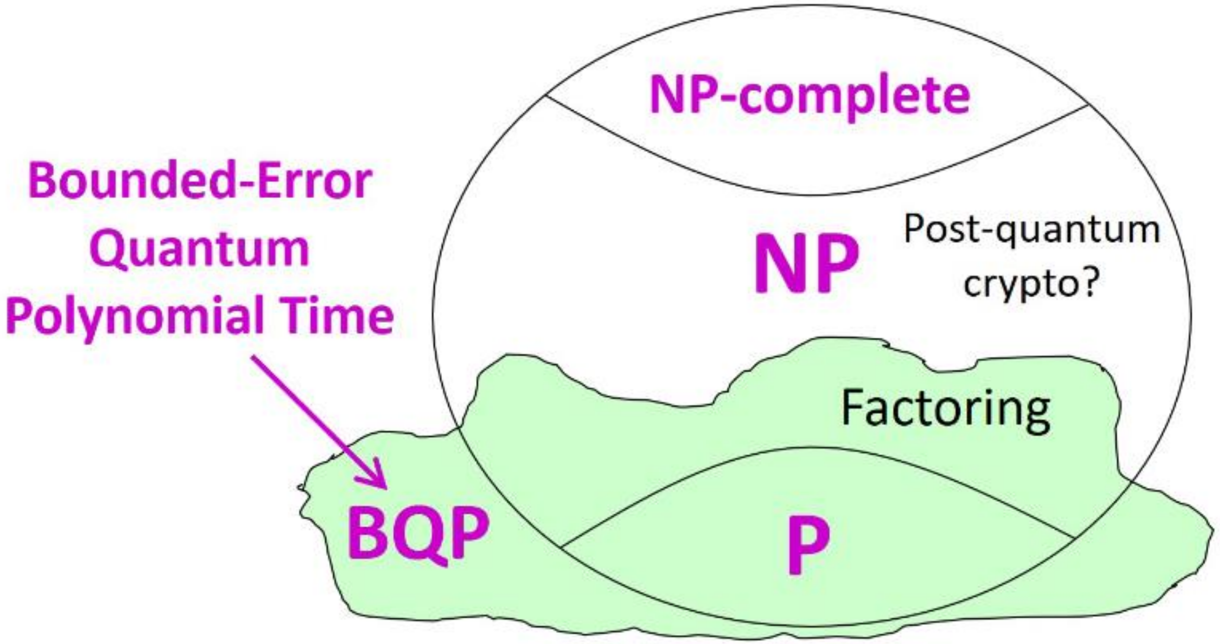
A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

Keywords: algorithmic number theory, prime factorization, discrete logarithms, Church's thesis, quantum computers, foundations of quantum mechanics, spin systems, Fourier transforms

AMS subject classifications: 81P10, 11Y05, 68Q10, 03D10

*A preliminary version of this paper appeared in the Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, Nov. 20–22, 1994, IEEE Computer Society Press, pp. 124–134.

†AT&T Research, Room 2D-149, 600 Mountain Ave., Murray Hill, NJ 07974.



Quantum Factoring

1996 — 2009 — 2018 — 2020

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

Keywords: algorithmic number theory, prime factorization, discrete logarithms, Church's thesis, quantum computers, foundations of quantum mechanics, spin systems, Fourier transforms

AMS subject classifications: 81P10, 11Y05, 68Q10, 03D10

*A preliminary version of this paper appeared in the Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, Nov. 20–22, 1994, IEEE Computer Society Press, pp. 124–134.

†AT&T Research, Room 2D-149, 600 Mountain Ave., Murray Hill, NJ 07974.

PRL **103,** 150502 (2009) PHYSICAL REVIEW LETTERS week ending 9 OCTOBER 2009

Quantum Algorithm for Linear Systems of Equations

Aram W. Harrow,[1] Avinatan Hassidim,[2] and Seth Lloyd[3]

[1]*Department of Mathematics, University of Bristol, Bristol, BS8 1TW, United Kingdom*
[2]*Research Laboratory for Electronics, MIT, Cambridge, Massachusetts 02139, USA*
[3]*Research Laboratory for Electronics and Department of Mechanical Engineering, MIT, Cambridge, Massachusetts 02139, USA*

(Received 5 July 2009; published 7 October 2009)

Solving linear systems of equations is a common problem that arises both on its own and as a subroutine in more complex problems: given a matrix A and a vector $\vec{b}$, find a vector $\vec{x}$ such that $A\vec{x} = \vec{b}$. We consider the case where one does not need to know the solution $\vec{x}$ itself, but rather an approximation of the expectation value of some operator associated with $\vec{x}$, e.g., $\vec{x}^\dagger M\vec{x}$ for some matrix M. In this case, when A is sparse, $N \times N$ and has condition number κ, the fastest known classical algorithms can find $\vec{x}$ and estimate $\vec{x}^\dagger M\vec{x}$ in time scaling roughly as $N\sqrt{\kappa}$. Here, we exhibit a quantum algorithm for estimating $\vec{x}^\dagger M\vec{x}$ whose runtime is a polynomial of $\log(N)$ and κ. Indeed, for small values of κ [i.e., poly $\log(N)$], we prove (using some common complexity-theoretic assumptions) that any classical algorithm for this problem generically requires exponentially more time than our quantum algorithm.

DOI: 10.1103/PhysRevLett.103.150502 PACS numbers: 03.67.Ac, 02.10.Ud, 89.70.Eg

Introduction.—Quantum computers are devices that harness quantum mechanics to perform computations in ways that classical computers cannot. For certain problems, quantum algorithms supply exponential speedups over their classical counterparts, the most famous example being Shor's factoring algorithm [1]. Few such exponential speedups are known, and those that are (such as the use of quantum computers to simulate other quantum systems [2]) have so far found limited use outside the domain of quantum mechanics. This Letter presents a quantum algorithm to estimate features of the solution of a set of linear equations. Compared to classical algorithms for the same task, our algorithm can be as much as exponentially faster.

Linear equations play an important role in virtually all fields of science and engineering. The sizes of the data sets that define the equations are growing rapidly over time, so that terabytes and even petabytes of data may need to be processed to obtain a solution. In other cases, such as when discretizing partial differential equations, the linear equations may be implicitly defined and thus far larger than the original description of the problem. For a classical computer, even to approximate the solution of Nlinear equations in N unknowns in general requires time that scales at least as N. Indeed, merely to write out the solution takes time of order N. Frequently, however, one is interested not in the full solution to the equations, but rather in computing some function of that solution, such as determining the total weight of some subset of the indices.

We show that in some cases, a quantum computer can approximate the value of such a function in time which scales logarithmically in N, and polynomially in the condition number (defined below) and desired precision. The dependence on N is exponentially better than what is achievable classically, while the dependence on condition number is comparable, and the dependence on error is worse. Typically, the accuracy required is not very large. However, the condition number often scales with the size of the problem, which presents a more serious limitation of our algorithm. Coping with large condition numbers has been studied extensively in the context of classical algorithms. In the discussion section, we will describe the applicability of some of the classical tools (pseudoinverses, preconditioners) to our quantum algorithm.

We sketch here the basic idea of our algorithm and then discuss it in more detail in the next section. Given a Hermitian $N \times N$ matrix A, and a unit vector $\vec{b}$, suppose we would like to find $\vec{x}$ satisfying $A\vec{x} = \vec{b}$. (We discuss later questions of efficiency as well as how the assumptions we have made about A and $\vec{b}$ can be relaxed.) First, the algorithm represents $\vec{b}$ as a quantum state $|b\rangle = \sum_{i=1}^N b_i|i\rangle$. Next, we use techniques of Hamiltonian simulation [3,4] to apply e^{iAt} to $|b\rangle$ for a superposition of different times t. This ability to exponentiate A translates, via the well-known technique of phase estimation [5,6], into the ability to decompose $|b\rangle$ in the eigenbasis of A and to find the corresponding eigenvalues λ_j. Informally, the state of the system after this stage is close to $\sum_{j=1}^N \beta_j|u_j\rangle|\lambda_j\rangle$, where u_j is the eigenvector basis of A, and $|b\rangle = \sum_{j=1}^N \beta_j|u_j\rangle$. We would then like to perform the linear map taking $|\lambda_j\rangle$ to $C\lambda_j^{-1}|\lambda_j\rangle$, where C is a normalizing constant. As this operation is not unitary, it has some probability of failing, which will enter into our discussion of the runtime below. After it succeeds, we uncompute the $|\lambda_j\rangle$ register and are left with a state proportional to $\sum_{j=1}^N \beta_j\lambda_j^{-1}|u_j\rangle = A^{-1}|b\rangle = |x\rangle$.

An important factor in the performance of the matrix inversion algorithm is κ, the condition number of A, or the ratio between A's largest and smallest eigenvalues. As the condition number grows, A becomes closer to a matrix which cannot be inverted, and the solutions become less stable. Our algorithms will generally assume that the sin-

0031-9007/09/103(15)/150502(4) 150502-1

Fast quantum (HHL) algorithm for a system of equations

1996 — 2009 — 2018 — 2020

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

Keywords: algorithmic number theory, prime factorization, discrete logarithms, Church's thesis, quantum computers, foundations of quantum mechanics, spin systems, Fourier transforms

AMS subject classifications: 81P10, 11Y05, 68Q10, 03D10

*A preliminary version of this paper appeared in the Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, Nov. 20–22, 1994, IEEE Computer Society Press, pp. 124–134.

†AT&T Research, Room 2D-149, 600 Mountain Ave., Murray Hill, NJ 07974.

PRL **103,** 150502 (2009) PHYSICAL REVIEW LETTERS week ending 9 OCTOBER 2009

Quantum Algorithm for Linear Systems of Equations

Aram W. Harrow,[1] Avinatan Hassidim,[2] and Seth Lloyd[3]

[1]*Department of Mathematics, University of Bristol, Bristol, BS8 1TW, United Kingdom*

[2]*Research Laboratory for Electronics, MIT, Cambridge, Massachusetts 02139, USA*

[3]*Research Laboratory for Electronics and Department of Mechanical Engineering, MIT, Cambridge, Massachusetts 02139, USA*

(Received 5 July 2009; published 7 October 2009)

Solving linear systems of equations is a common problem that arises both on its own and as a subroutine in more complex problems: given a matrix A and a vector $\vec{b}$, find a vector $\vec{x}$ such that $A\vec{x} = \vec{b}$. We consider the case where one does not need to know the solution $\vec{x}$ itself, but rather an approximation of the expectation value of some operator associated with $\vec{x}$, e.g., $\vec{x}^\dagger M\vec{x}$ for some matrix M. In this case, when A is sparse, $N \times N$ and has condition number κ, the fastest known classical algorithms can find $\vec{x}$ and estimate $\vec{x}^\dagger M\vec{x}$ in time scaling roughly as $N\sqrt{\kappa}$. Here, we exhibit a quantum algorithm for estimating $\vec{x}^\dagger M\vec{x}$ whose runtime is a polynomial of $\log(N)$ and κ. Indeed, for small values of κ [i.e., poly $\log(N)$], we prove (using some common complexity-theoretic assumptions) that any classical algorithm for this problem generically requires exponentially more time than our quantum algorithm.

DOI: 10.1103/PhysRevLett.103.150502 PACS numbers: 03.67.Ac, 02.10.Ud, 89.70.Eg

Introduction.—Quantum computers are devices that harness quantum mechanics to perform computations in ways that classical computers cannot. For certain problems, quantum algorithms supply exponential speedups over their classical counterparts, the most famous example being Shor's factoring algorithm [1]. Few such exponential speedups are known, and those that are (such as the use of quantum computers to simulate other quantum systems [2]) have so far found limited use outside the domain of quantum mechanics. This Letter presents a quantum algorithm to estimate features of the solution of a set of linear equations. Compared to classical algorithms for the same task, our algorithm can be as much as exponentially faster.

Linear equations play an important role in virtually all fields of science and engineering. The sizes of the data sets that define the equations are growing rapidly over time, so that terabytes and even petabytes of data may need to be processed to obtain a solution. In other cases, such as when discretizing partial differential equations, the linear equations may be implicitly defined and thus far larger than the original description of the problem. For a classical computer, even to approximate the solution of Nlinear equations in N unknowns in general requires time that scales at least as N. Indeed, merely to write out the solution takes time of order N. Frequently, however, one is interested not in the full solution to the equations, but rather in computing some function of that solution, such as determining the total weight of some subset of the indices.

We show that in some cases, a quantum computer can approximate the value of such a function in time which scales logarithmically in N, and polynomially in the condition number (defined below) and desired precision. The dependence on N is exponentially better than what is achievable classically, while the dependence on condition number is comparable, and the dependence on error is worse. Typically, the accuracy required is not very large. However, the condition number often scales with the size of the problem, which presents a more serious limitation of our algorithm. Coping with large condition numbers has been studied extensively in the context of classical algorithms. In the discussion section, we will describe the applicability of some of the classical tools (pseudoinverses, preconditioners) to our quantum algorithm.

We sketch here the basic idea of our algorithm and then discuss it in more detail in the next section. Given a Hermitian $N \times N$ matrix A, and a unit vector $\vec{b}$, suppose we would like to find $\vec{x}$ satisfying $A\vec{x} = \vec{b}$. (We discuss later questions of efficiency as well as how the assumptions we have made about A and $\vec{b}$ can be relaxed.) First, the algorithm represents $\vec{b}$ as a quantum state $|b\rangle = \sum_{i=1}^N b_i|i\rangle$. Next, we use techniques of Hamiltonian simulation [3,4] to apply e^{iAt} to $|b\rangle$ for a superposition of different times t. This ability to exponentiate A translates, via the well-known technique of phase estimation [5,6], into the ability to decompose $|b\rangle$ in the eigenbasis of A and to find the corresponding eigenvalues λ_j. Informally, the state of the system after this stage is close to $\sum_{j=1}^N \beta_j|u_j\rangle|\lambda_j\rangle$, where u_j is the eigenvector basis of A, and $|b\rangle = \sum_{j=1}^N \beta_j|u_j\rangle$. We would then like to perform the linear map taking $|\lambda_j\rangle$ to $C\lambda_j^{-1}|\lambda_j\rangle$, where C is a normalizing constant. As this operation is not unitary, it has some probability of failing, which will enter into our discussion of the runtime below. After it succeeds, we uncompute the $|\lambda_j\rangle$ register and are left with a state proportional to $\sum_{j=1}^N \beta_j\lambda_j^{-1}|u_j\rangle = A^{-1}|b\rangle = |x\rangle$.

An important factor in the performance of the matrix inversion algorithm is κ, the condition number of A, or the ratio between A's largest and smallest eigenvalues. As the condition number grows, A becomes closer to a matrix which cannot be inverted, and the solutions become less stable. Our algorithms will generally assume that the sin-

MIT-CTP/4610

A Quantum Approximate Optimization Algorithm

Edward Farhi and Jeffrey Goldstone

Center for Theoretical Physics

Massachusetts Institute of Technology

Cambridge, MA 02139

Sam Gutmann

Abstract

We introduce a quantum algorithm that produces approximate solutions for combinatorial optimization problems. The algorithm depends on an integer $p \geq 1$ and the quality of the approximation improves as p is increased. The quantum circuit that implements the algorithm consists of unitary gates whose locality is at most the locality of the objective function whose optimum is sought. The depth of the circuit grows linearly with p times (at worst) the number of constraints. If p is fixed, that is, independent of the input size, the algorithm makes use of efficient classical preprocessing. If p grows with the input size a different strategy is proposed. We study the algorithm as applied to MaxCut on regular graphs and analyze its performance on 2-regular and 3-regular graphs for fixed p. For $p = 1$, on 3-regular graphs the quantum algorithm always finds a cut that is at least 0.6924 times the size of the optimal cut.

Quantum Variational Algorithm (QVA)

1996 2009 2015 2020

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

Keywords: algorithmic number theory, prime factorization, discrete logarithms, Church's thesis, quantum computers, foundations of quantum mechanics, spin systems, Fourier transforms

AMS subject classifications: 81P10, 11Y05, 68Q10, 03D10

*A preliminary version of this paper appeared in the Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, Nov. 20–22, 1994, IEEE Computer Society Press, pp. 124–134.

†AT&T Research, Room 2D-149, 600 Mountain Ave., Murray Hill, NJ 07974.

PRL **103,** 150502 (2009) PHYSICAL REVIEW LETTERS week ending 9 OCTOBER 2009

Quantum Algorithm for Linear Systems of Equations

Aram W. Harrow,[1] Avinatan Hassidim,[2] and Seth Lloyd[3]

[1]*Department of Mathematics, University of Bristol, Bristol, BS8 1TW, United Kingdom*
[2]*Research Laboratory for Electronics, MIT, Cambridge, Massachusetts 02139, USA*
[3]*Research Laboratory for Electronics and Department of Mechanical Engineering, MIT, Cambridge, Massachusetts 02139, USA*

(Received 5 July 2009; published 7 October 2009)

Solving linear systems of equations is a common problem that arises both on its own and as a subroutine in more complex problems: given a matrix A and a vector $\vec{b}$, find a vector $\vec{x}$ such that $A\vec{x} = \vec{b}$. We consider the case where one does not need to know the solution $\vec{x}$ itself, but rather an approximation of the expectation value of some operator associated with $\vec{x}$, e.g., $\vec{x}^\dagger M\vec{x}$ for some matrix M. In this case, when A is sparse, $N \times N$ and has condition number κ, the fastest known classical algorithms can find $\vec{x}$ and estimate $\vec{x}^\dagger M\vec{x}$ in time scaling roughly as $N\sqrt{\kappa}$. Here, we exhibit a quantum algorithm for estimating $\vec{x}^\dagger M\vec{x}$ whose runtime is a polynomial of $\log(N)$ and κ. Indeed, for small values of κ [i.e., poly $\log(N)$], we prove (using some common complexity-theoretic assumptions) that any classical algorithm for this problem generically requires exponentially more time than our quantum algorithm.

DOI: 10.1103/PhysRevLett.103.150502 PACS numbers: 03.67.Ac, 02.10.Ud, 89.70.Eg

Introduction.—Quantum computers are devices that harness quantum mechanics to perform computations in ways that classical computers cannot. For certain problems, quantum algorithms supply exponential speedups over their classical counterparts, the most famous example being Shor's factoring algorithm [1]. Few such exponential speedups are known, and those that are (such as the use of quantum computers to simulate other quantum systems [2]) have so far found limited use outside the domain of quantum mechanics. This Letter presents a quantum algorithm to estimate features of the solution of a set of linear equations. Compared to classical algorithms for the same task, our algorithm can be as much as exponentially faster.

Linear equations play an important role in virtually all fields of science and engineering. The sizes of the data sets that define the equations are growing rapidly over time, so that terabytes and even petabytes of data may need to be processed to obtain a solution. In other cases, such as when discretizing partial differential equations, the linear equations may be implicitly defined and thus far larger than the original description of the problem. For a classical computer, even to approximate the solution of Nlinear equations in N unknowns in general requires time that scales at least as N. Indeed, merely to write out the solution takes time of order N. Frequently, however, one is interested not in the full solution to the equations, but rather in computing some function of that solution, such as determining the total weight of some subset of the indices.

We show that in some cases, a quantum computer can approximate the value of such a function in time which scales logarithmically in N, and polynomially in the condition number (defined below) and desired precision. The dependence on N is exponentially better than what is achievable classically, while the dependence on condition number is comparable, and the dependence on error is worse. Typically, the accuracy required is not very large. However, the condition number often scales with the size of the problem, which presents a more serious limitation of our algorithm. Coping with large condition numbers has been studied extensively in the context of classical algorithms. In the discussion section, we will describe the applicability of some of the classical tools (pseudoinverses, preconditioners) to our quantum algorithm.

We sketch here the basic idea of our algorithm and then discuss it in more detail in the next section. Given a Hermitian $N \times N$ matrix A, and a unit vector $\vec{b}$, suppose we would like to find $\vec{x}$ satisfying $A\vec{x} = \vec{b}$. (We discuss later questions of efficiency as well as how the assumptions we have made about A and $\vec{b}$ can be relaxed.) First, the algorithm represents $\vec{b}$ as a quantum state $|b\rangle = \sum_{i=1}^{N} b_i|i\rangle$. Next, we use techniques of Hamiltonian simulation [3,4] to apply e^{iAt} to $|b\rangle$ for a superposition of different times t. This ability to exponentiate A translates, via the well-known technique of phase estimation [5,6], into the ability to decompose $|b\rangle$ in the eigenbasis of A and to find the corresponding eigenvalues λ_j. Informally, the state of the system after this stage is close to $\sum_{j=1}^{N} \beta_j|u_j\rangle|\lambda_j\rangle$, where u_j is the eigenvector basis of A, and $|b\rangle = \sum_{j=1}^{N} \beta_j|u_j\rangle$. We would then like to perform the linear map taking $|\lambda_j\rangle$ to $C\lambda_j^{-1}|\lambda_j\rangle$, where C is a normalizing constant. As this operation is not unitary, it has some probability of failing, which will enter into our discussion of the runtime below. After it succeeds, we uncompute the $|\lambda_j\rangle$ register and are left with a state proportional to $\sum_{j=1}^{N} \beta_j\lambda_j^{-1}|u_j\rangle = A^{-1}|b\rangle = |x\rangle$.

An important factor in the performance of the matrix inversion algorithm is κ, the condition number of A, or the ratio between A's largest and smallest eigenvalues. As the condition number grows, A becomes closer to a matrix which cannot be inverted, and the solutions become less stable. Our algorithms will generally assume that the sin-

0031-9007/09/103(15)/150502(4) 150502-1

MIT-CTP/4610

A Quantum Approximate Optimization Algorithm

Edward Farhi and Jeffrey Goldstone
Center for Theoretical Physics
Massachusetts Institute of Technology
Cambridge, MA 02139

Sam Gutmann

Abstract

We introduce a quantum algorithm that produces approximate solutions for combinatorial optimization problems. The algorithm depends on an integer $p \geq 1$ and the quality of the approximation improves as p is increased. The quantum circuit that implements the algorithm consists of unitary gates whose locality is at most the locality of the objective function whose optimum is sought. The depth of the circuit grows linearly with p times (at worst) the number of constraints. If p is fixed, that is, independent of the input size, the algorithm makes use of efficient classical preprocessing. If p grows with the input size a different strategy is proposed. We study the algorithm as applied to MaxCut on regular graphs and analyze its performance on 2-regular and 3-regular graphs for fixed p. For $p = 1$, on 3-regular graphs the quantum algorithm always finds a cut that is at least 0.6924 times the size of the optimal cut.

The power of quantum neural networks

Amira Abbas[1,2], David Sutter[1], Christa Zoufal[1,3], Aurelien Lucchi[3], Alessio Figalli[3] and Stefan Woerner[1]✉

It is unknown whether near-term quantum computers are advantageous for machine learning tasks. In this work we address this question by trying to understand how powerful and trainable quantum machine learning models are in relation to popular classical neural networks. We propose the effective dimension—a measure that captures these qualities—and prove that it can be used to assess any statistical model's ability to generalize on new data. Crucially, the effective dimension is a data-dependent measure that depends on the Fisher information, which allows us to gauge the ability of a model to train. We demonstrate numerically that a class of quantum neural networks is able to achieve a considerably better effective dimension than comparable feedforward networks and train faster, suggesting an advantage for quantum machine learning, which we verify on real quantum hardware.

The power of a model lies in its ability to fit a variety of functions[1]. In machine learning, power is often referred to as a model's capacity to express different relationships between variables[2]. Deep neural networks have proven to be extremely powerful models, capable of capturing intricate relationships by learning from data[3]. Quantum neural networks serve as a newer class of machine learning models that are deployed on quantum computers and use quantum effects such as superposition, entanglement and interference to perform computation. Some proposals for quantum neural networks include[4–11]—and hint at—potential advantages such as speed-ups in training and faster processing. Although there has been much development in the growing field of quantum machine learning, a systematic study of the trade-offs between quantum and classical models has yet to be conducted[12]. In particular, the question of whether quantum neural networks are more powerful than classical neural networks is still open.

A common way to quantify the power of a model is by its complexity[13]. In statistical learning theory, the Vapnik–Chervonenkis dimension is an established complexity measure, where error bounds on how well a model generalizes (that is, performs on unseen data) can be derived[14]. Although the Vapnik–Chervonenkis dimension has attractive properties in theory, computing it in practice is notoriously difficult. Furthermore, using the Vapnik–Chervonenkis dimension to bound generalization error requires several unrealistic assumptions, including that the model has access to infinite data[15,16]. The measure also scales with the number of parameters in the model and ignores the distribution of data. As modern deep neural networks are heavily overparameterized, generalization bounds based on the Vapnik–Chervonenkis dimension—and other measures alike—are typically vacuous[17,18].

In ref. [19], the authors analyzed the expressive power of parameterized quantum circuits using memory capacity and found that quantum neural networks had limited advantages over classical neural networks. Memory capacity is, however, closely related to the Vapnik–Chervonenkis dimension and is thus subject to similar criticisms. In ref. [20], a quantum neural network is presented that exhibits a higher expressibility than certain classical models, captured by the types of probability distributions it can generate. Another result from ref. [21] is based on strong heuristics and provides systematic examples of possible advantages for quantum neural networks.

We turn our attention to measures that are easy to estimate in practice and, importantly, incorporate the distribution of data. In particular, measures such as the effective dimension have been motivated from an information-theoretic standpoint and depend on the Fisher information, a quantity that describes the geometry of a model's parameter space and is essential in both statistics and machine learning[22–24]. We argue that the effective dimension is a robust capacity measure through proof of a generalization error bound and supporting numerical analyses, and thus use this measure to study the power of a popular class of neural networks in both classical and quantum regimes.

Despite a lack of quantitative statements on the power of quantum neural networks, another issue is rooted in the trainability of these models. A precise connection between expressibility and trainability for certain classes of quantum neural networks is outlined in refs. [25,26]. Quantum neural networks often suffer from the barren plateau phenomenon, wherein the loss landscape is perilously flat and parameter optimization is therefore extremely difficult[27]. As shown in ref. [28], barren plateaus may be noise induced, where certain noise models are assumed on the hardware. In other words, the effect of hardware noise can make it very difficult to train a quantum model. Furthermore, barren plateaus can be circuit induced, which relates to the design of a model and random parameter initialization. Methods to avoid the latter have been explored in refs. [29–32], but noise-induced barren plateaus remain problematic.

A particular attempt to understand the loss landscape of quantum models uses the Hessian[33], which quantifies the curvature of a model's loss function at a point in its parameter space[34]. Properties of the Hessian, such as its spectrum, provide useful diagnostic information on the trainability of a model[35]. It was discovered that the entries of the Hessian vanish exponentially in models suffering from a barren plateau[36]. For certain loss functions, the Fisher information matrix coincides with the Hessian of the loss function[37]. Consequently, we can examine the trainability of quantum and classical neural networks by analyzing the Fisher information matrix, which is incorporated by the effective dimension. In this way, we may explicitly relate the effective dimension to model trainability[38].

We find that a class of quantum neural networks is able to achieve a considerably higher capacity and faster training ability numerically than comparable classical neural networks. A higher capacity is captured by a higher effective dimension, whereas

[1]IBM Quantum, IBM Research—Zurich, Rueschlikon, Switzerland. [2]University of KwaZulu-Natal, Durban, South Africa. [3]ETH Zurich, Zurich, Switzerland.
✉e-mail: wor@zurich.ibm.com

Quantum Factoring	Fast HHL algorithm	Quantum Variational Algorithm	Power of quantum variational algorithms
1996	2009	2015	2020

Fault-tolerant quantum computers

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor[†]

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

Keywords: algorithmic number theory, prime factorization, discrete logarithms, Church's thesis, quantum computers, foundations of quantum mechanics, spin systems, Fourier transforms

AMS subject classifications: 81P10, 11Y05, 68Q10, 03D10

*A preliminary version of this paper appeared in the Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, Nov. 20–22, 1994, IEEE Computer Society Press, pp. 124–134.

[†]AT&T Research, Room 2D-149, 600 Mountain Ave., Murray Hill, NJ 07974.

PRL 103, 150502 (2009) PHYSICAL REVIEW LETTERS week ending 9 OCTOBER 2009

Quantum Algorithm for Linear Systems of Equations

Aram W. Harrow,[1] Avinatan Hassidim,[2] and Seth Lloyd[3]

[1]Department of Mathematics, University of Bristol, Bristol, BS8 1TW, United Kingdom
[2]Research Laboratory for Electronics, MIT, Cambridge, Massachusetts 02139, USA
[3]Research Laboratory for Electronics and Department of Mechanical Engineering, MIT, Cambridge, Massachusetts 02139, USA

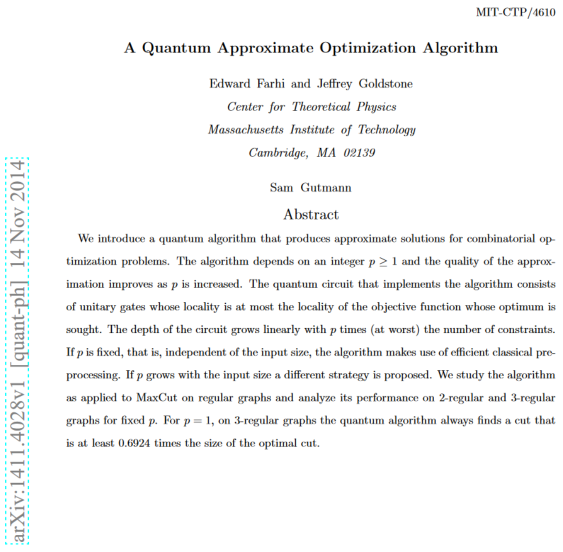
(Received 5 July 2009; published 7 October 2009)

Solving linear systems of equations is a common problem that arises both on its own and as a subroutine in more complex problems: given a matrix A and a vector $\vec{b}$, find a vector $\vec{x}$ such that $A\vec{x} = \vec{b}$. We consider the case where one does not need to know the solution $\vec{x}$ itself, but rather an approximation of the expectation value of some operator associated with $\vec{x}$, e.g., $\vec{x}^\dagger M \vec{x}$ for some matrix M. In this case, when A is sparse, $N \times N$ and has condition number κ, the fastest known classical algorithms can find $\vec{x}$ and estimate $\vec{x}^\dagger M \vec{x}$ in time scaling roughly as $N\sqrt{\kappa}$. Here, we exhibit a quantum algorithm for estimating $\vec{x}^\dagger M \vec{x}$ whose runtime is a polynomial of $\log(N)$ and κ. Indeed, for small values of κ [i.e., poly $\log(N)$], we prove (using some common complexity-theoretic assumptions) that any classical algorithm for this problem generically requires exponentially more time than our quantum algorithm.

DOI: 10.1103/PhysRevLett.103.150502 PACS numbers: 03.67.Ac, 02.10.Ud, 89.70.Eg

0031-9007/09/103(15)/150502(4) 150502-1

Noisy-intermediate scale quantum computers



MIT-CTP/4610

A Quantum Approximate Optimization Algorithm

Edward Farhi and Jeffrey Goldstone
Center for Theoretical Physics
Massachusetts Institute of Technology
Cambridge, MA 02139

Sam Gutmann

Abstract

We introduce a quantum algorithm that produces approximate solutions for combinatorial optimization problems. The algorithm depends on an integer $p \geq 1$ and the quality of the approximation improves as p is increased. The quantum circuit that implements the algorithm consists of unitary gates whose locality is at most the locality of the objective function whose optimum is sought. The depth of the circuit grows linearly with p times (at worst) the number of constraints. If p is fixed, that is, independent of the input size, the algorithm makes use of efficient classical preprocessing. If p grows with the input size a different strategy is proposed. We study the algorithm as applied to MaxCut on regular graphs and analyze its performance on 2-regular and 3-regular graphs for fixed p. For $p = 1$, on 3-regular graphs the quantum algorithm always finds a cut that is at least 0.6924 times the size of the optimal cut.

The power of quantum neural networks

Amira Abbas[1,2], David Sutter[1], Christa Zoufal[1,3], Aurelien Lucchi[3], Alessio Figalli[3] and Stefan Woerner[1]

It is unknown whether near-term quantum computers are advantageous for machine learning tasks. In this work we address this question by trying to understand how powerful and trainable quantum machine learning models are in relation to popular classical neural networks. We propose the effective dimension—a measure that captures these qualities—and prove that it can be used to assess any statistical model's ability to generalize on new data. Crucially, the effective dimension is a data-dependent measure that depends on the Fisher information, which allows us to gauge the ability of a model to train. We demonstrate numerically that a class of quantum neural networks is able to achieve a considerably better effective dimension than comparable feedforward networks and train faster, suggesting an advantage for quantum machine learning, which we verify on real quantum hardware.

[1]IBM Quantum, IBM Research—Zurich, Rueschlikon, Switzerland. [2]University of KwaZulu-Natal, Durban, South Africa. [3]ETH Zurich, Zurich, Switzerland. e-mail: wor@zurich.ibm.com

Quantum Factoring — 1996

Fast HHL algorithm — 2009

Quantum Variational Algorithm (QVA) — 2015

Power of QVA — 2020

Noisy-intermediate scale quantum (NISQ) computers

MIT-CTP/4610

A Quantum Approximate Optimization Algorithm

Edward Farhi and Jeffrey Goldstone
Center for Theoretical Physics
Massachusetts Institute of Technology
Cambridge, MA 02139

Sam Gutmann

arXiv:1411.4028v1 [quant-ph] 14 Nov 2014

Abstract

We introduce a quantum algorithm that produces approximate solutions for combinatorial optimization problems. The algorithm depends on an integer $p \geq 1$ and the quality of the approximation improves as p is increased. The quantum circuit that implements the algorithm consists of unitary gates whose locality is at most the locality of the objective function whose optimum is sought. The depth of the circuit grows linearly with p times (at worst) the number of constraints. If p is fixed, that is, independent of the input size, the algorithm makes use of efficient classical preprocessing. If p grows with the input size a different strategy is proposed. We study the algorithm as applied to MaxCut on regular graphs and analyze its performance on 2-regular and 3-regular graphs for fixed p. For $p = 1$, on 3-regular graphs the quantum algorithm always finds a cut that is at least 0.6924 times the size of the optimal cut.

The power of quantum neural networks

Amira Abbas[1,2], David Sutter[1], Christa Zoufal[1,3], Aurelien Lucchi[3], Alessio Figalli[3] and Stefan Woerner[1]✉

It is unknown whether near-term quantum computers are advantageous for machine learning tasks. In this work we address this question by trying to understand how powerful and trainable quantum machine learning models are in relation to popular classical neural networks. We propose the effective dimension—a measure that captures these qualities—and prove that it can be used to assess any statistical model's ability to generalize on new data. Crucially, the effective dimension is a data-dependent measure that depends on the Fisher information, which allows us to gauge the ability of a model to train. We demonstrate numerically that a class of quantum neural networks is able to achieve a considerably better effective dimension than comparable feedforward networks and train faster, suggesting an advantage for quantum machine learning, which we verify on real quantum hardware.

The power of a model lies in its ability to fit a variety of functions[1]. In machine learning, power is often referred to as a model's capacity to express different relationships between variables[2]. Deep neural networks have proven to be extremely powerful models, capable of capturing intricate relationships by learning from data[3]. Quantum neural networks serve as a newer class of machine learning models that are deployed on quantum computers and use quantum effects such as superposition, entanglement and interference to perform computation. Some proposals for quantum neural networks include[4–11]—and hint at—potential advantages such as speed-ups in training and faster processing. Although there has been much development in the growing field of quantum machine learning, a systematic study of the trade-offs between quantum and classical models has yet to be conducted[12]. In particular, the question of whether quantum neural networks are more powerful than classical neural networks is still open.

A common way to quantify the power of a model is by its complexity[13]. In statistical learning theory, the Vapnik–Chervonenkis dimension is an established complexity measure, where error bounds on how well a model generalizes (that is, performs on unseen data) can be derived[14]. Although the Vapnik–Chervonenkis dimension has attractive properties in theory, computing it in practice is notoriously difficult. Furthermore, using the Vapnik–Chervonenkis dimension to bound generalization error requires several unrealistic assumptions, including that the model has access to infinite data[15,16]. The measure also scales with the number of parameters in the model and ignores the distribution of data. As modern deep neural networks are heavily overparameterized, generalization bounds based on the Vapnik–Chervonenkis dimension—and other measures alike—are typically vacuous[17,18].

In ref. [19], the authors analyzed the expressive power of parameterized quantum circuits using memory capacity and found that quantum neural networks had limited advantages over classical neural networks. Memory capacity is, however, closely related to the Vapnik–Chervonenkis dimension and is thus subject to similar criticisms. In ref. [20], a quantum neural network is presented that exhibits a higher expressibility than certain classical models, captured by the types of probability distributions it can generate. Another result from ref. [21] is based on strong heuristics and provides systematic examples of possible advantages for quantum neural networks.

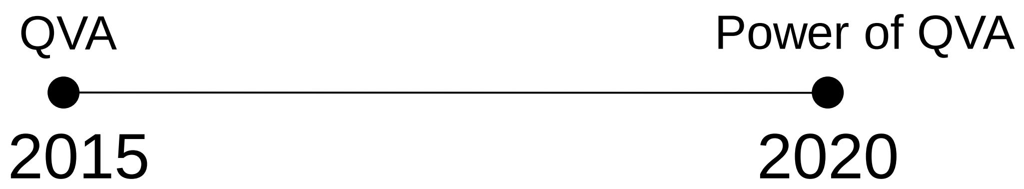
We turn our attention to measures that are easy to estimate in practice and, importantly, incorporate the distribution of data. In particular, measures such as the effective dimension have been motivated from an information-theoretic standpoint and depend on the Fisher information, a quantity that describes the geometry of a model's parameter space and is essential in both statistics and machine learning[22–24]. We argue that the effective dimension is a robust capacity measure through proof of a generalization error bound and supporting numerical analyses, and thus use this measure to study the power of a popular class of neural networks in both classical and quantum regimes.

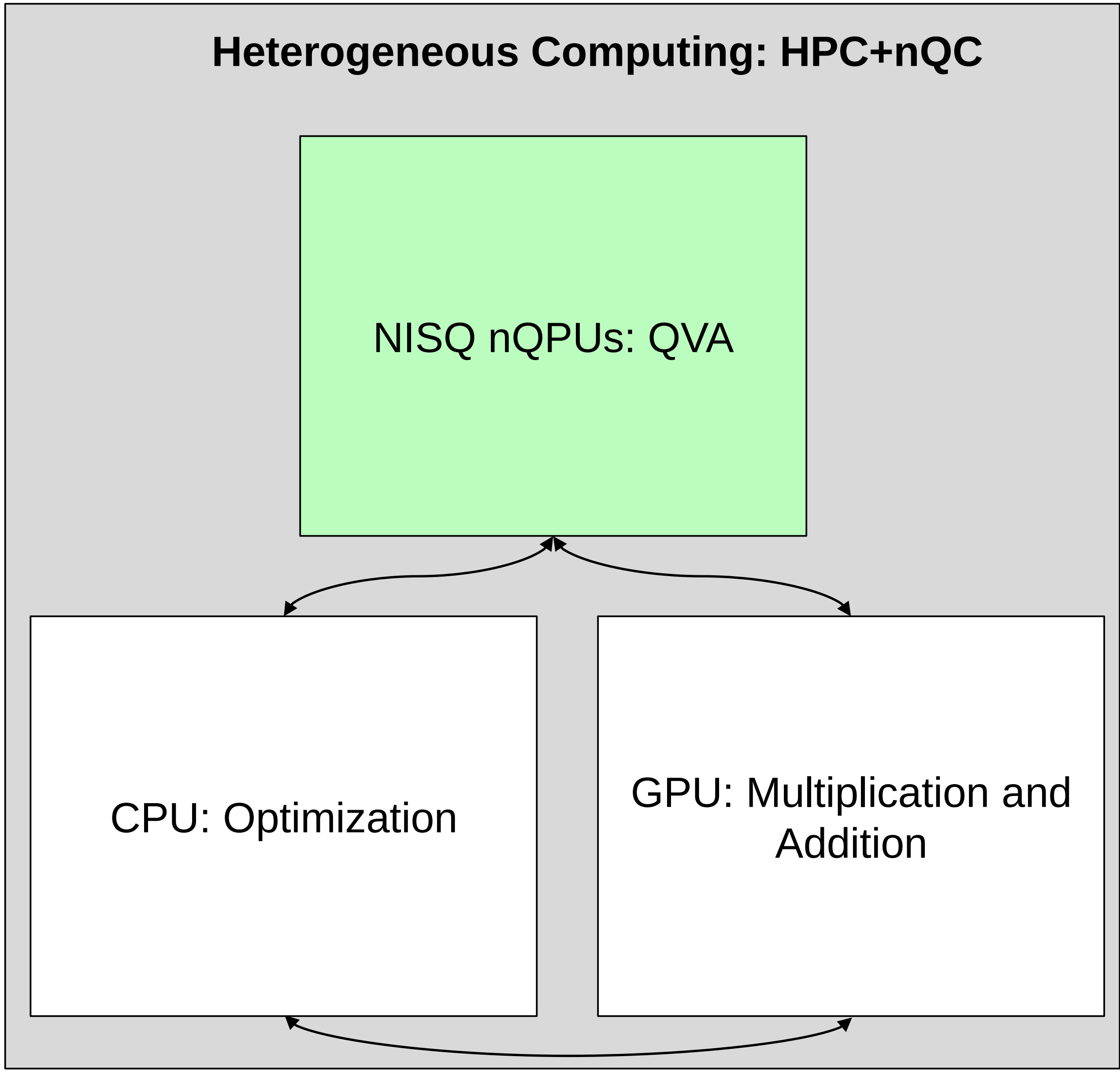
Despite a lack of quantitative statements on the power of quantum neural networks, another issue is rooted in the trainability of these models. A precise connection between expressibility and trainability for certain classes of quantum neural networks is outlined in refs. [25,26]. Quantum neural networks often suffer from the barren plateau phenomenon, wherein the loss landscape is perilously flat and parameter optimization is therefore extremely difficult[27]. As shown in ref. [28], barren plateaus may be noise induced, where certain noise models are assumed on the hardware. In other words, the effect of hardware noise can make it very difficult to train a quantum model. Furthermore, barren plateaus can be circuit induced, which relates to the design of a model and random parameter initialization. Methods to avoid the latter have been explored in refs. [29–32], but noise-induced barren plateaus remain problematic.

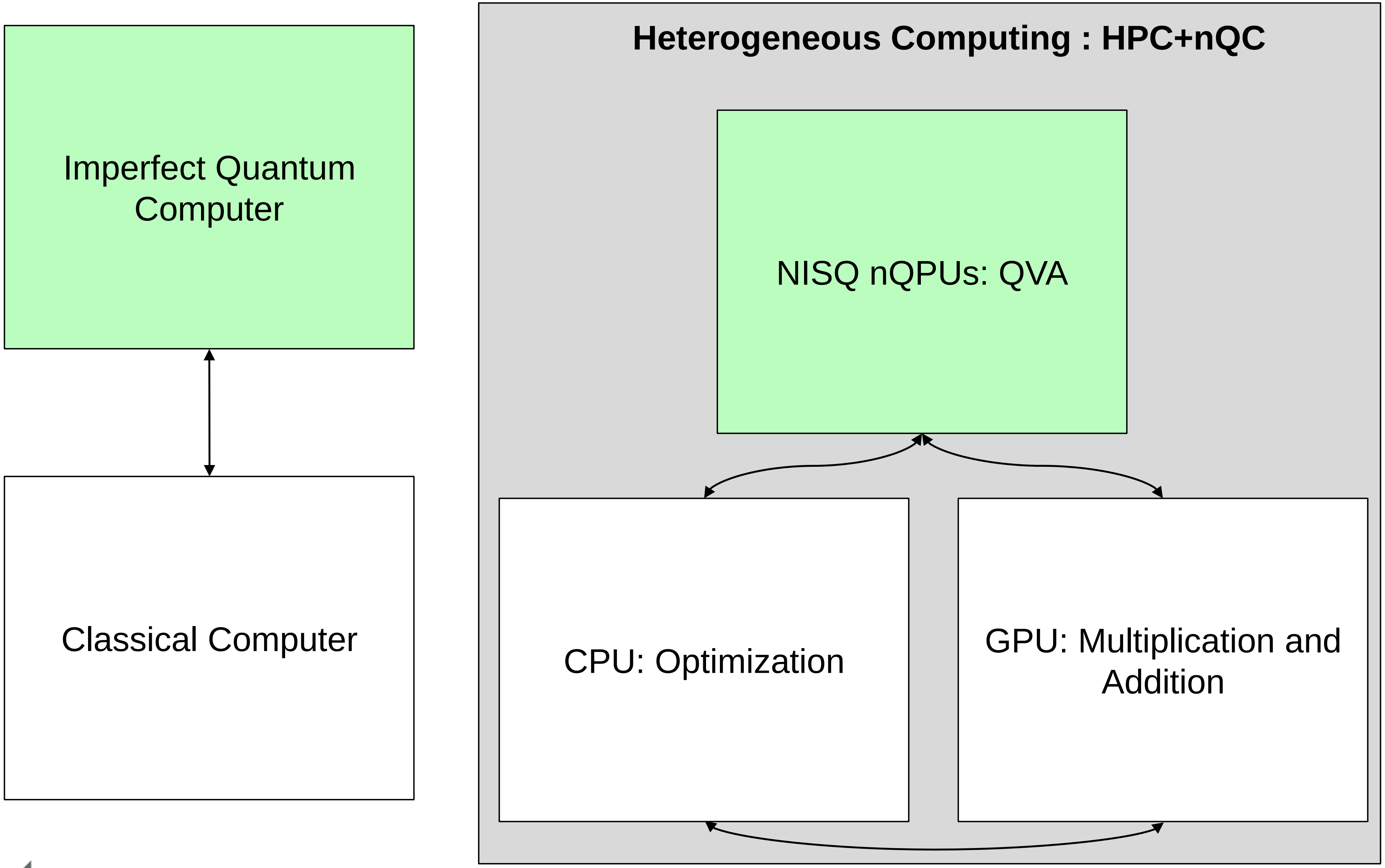
A particular attempt to understand the loss landscape of quantum models uses the Hessian[33], which quantifies the curvature of a model's loss function at a point in its parameter space[34]. Properties of the Hessian, such as its spectrum, provide useful diagnostic information on the trainability of a model[35]. It was discovered that the entries of the Hessian vanish exponentially in models suffering from a barren plateau[36]. For certain loss functions, the Fisher information matrix coincides with the Hessian of the loss function[37]. Consequently, we can examine the trainability of quantum and classical neural networks by analyzing the Fisher information matrix, which is incorporated by the effective dimension. In this way, we may explicitly relate the effective dimension to model trainability[38].

We find that a class of quantum neural networks is able to achieve a considerably higher capacity and faster training ability numerically than comparable classical feedforward neural networks. A higher capacity is captured by a higher effective dimension, whereas

[1]IBM Quantum, IBM Research—Zurich, Rueschlikon, Switzerland. [2]University of KwaZulu-Natal, Durban, South Africa. [3]ETH Zurich, Zurich, Switzerland.
✉e-mail: wor@zurich.ibm.com



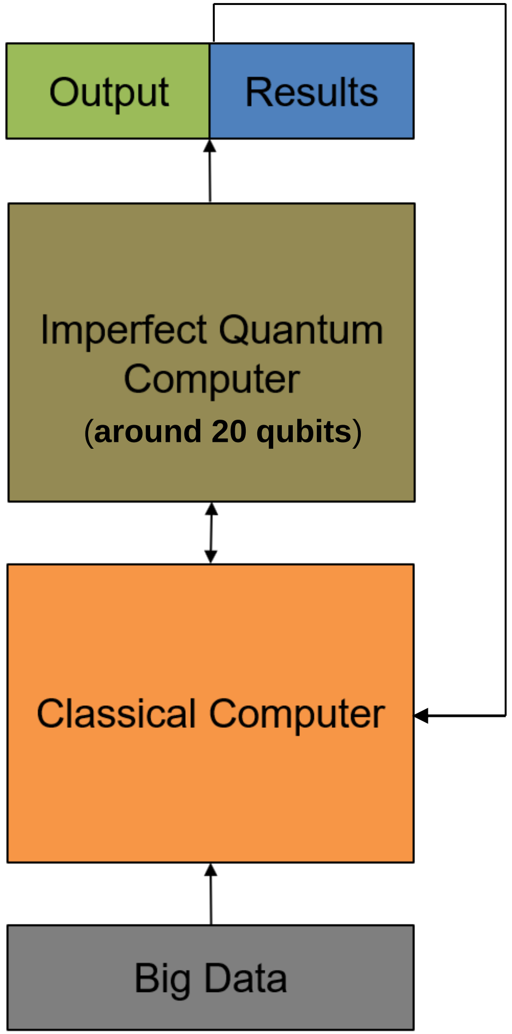


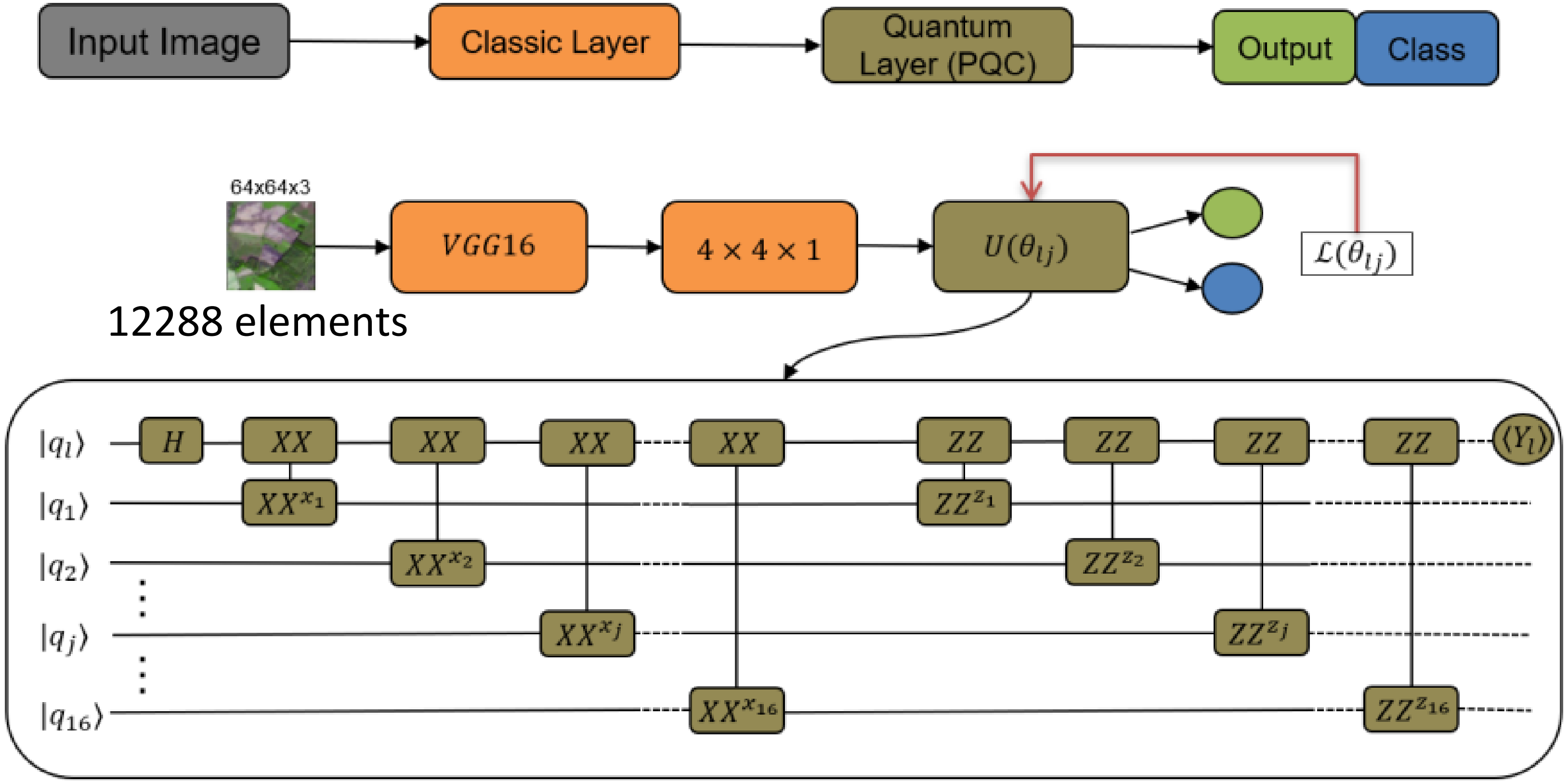


nQC (or nQPUs): superconducting, neural atomic, photonic quantum computers, or quantum annealer

Lets forget about quantum advantage. BUT What is exactly a quantum computer and How to make it work for machine learning tasks or for processing big datasets in Earth Observation?

Quantum Variational Algorithm for Earth Observation: Case I

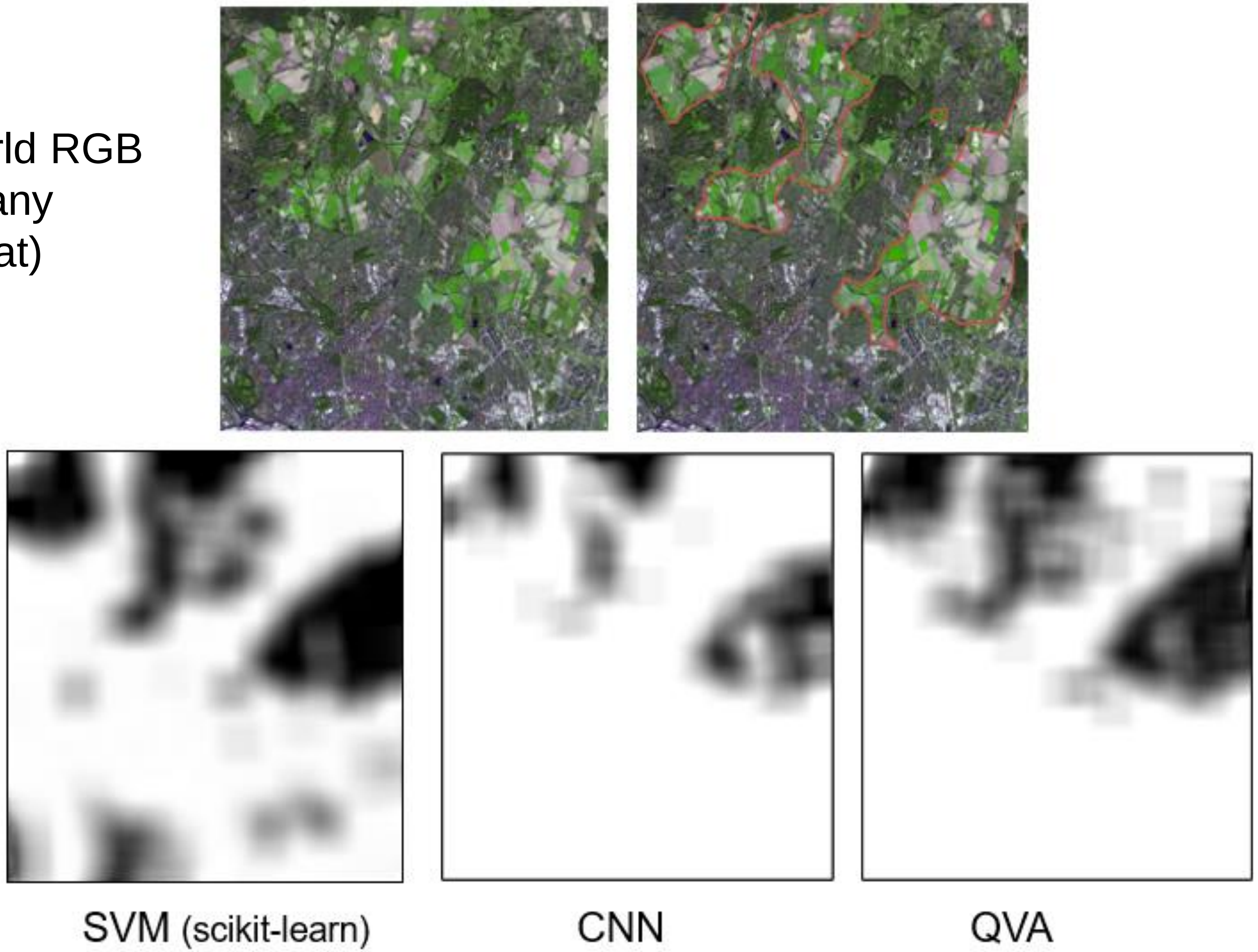




S. Otgonbaatar and M. Datcu, "Classification of Remote Sensing Images With Parameterized Quantum Gates," in *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1-5, 2022, Art no. 8020105, doi: 10.1109/LGRS.2021.3108014.

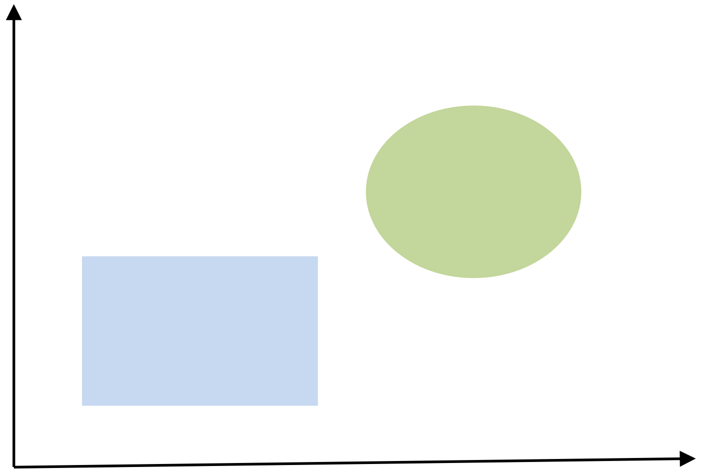
Quantum Variational Algorithm for Earth Observation: Case I

Test QVA on a real-world RGB image of Berlin, Germany (trained QVA on Eurosat)



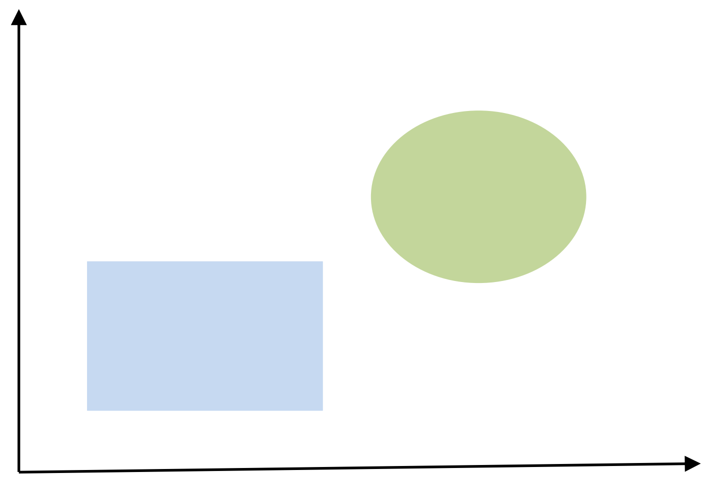
Quantum Variational Algorithm for Earth Observation: Case II

Data sets: different distribution



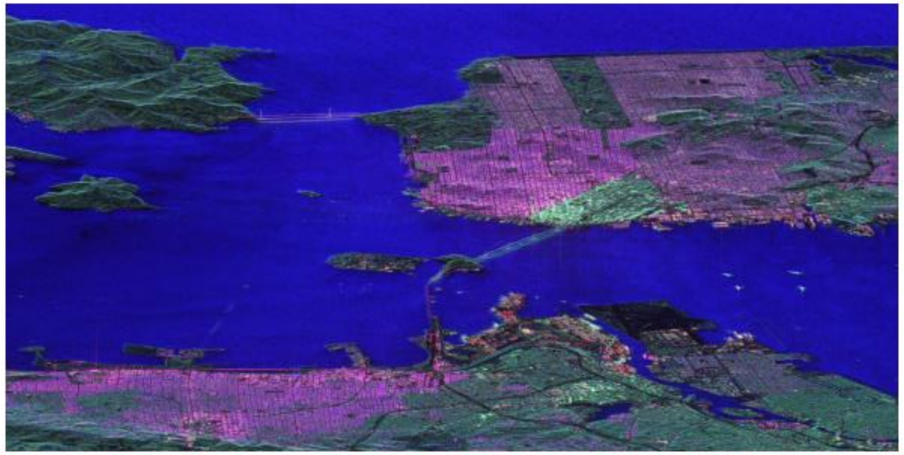
out-of-distribution? → **physics** to rescue

Quantum Variational Algorithm for Earth Observation: Case II

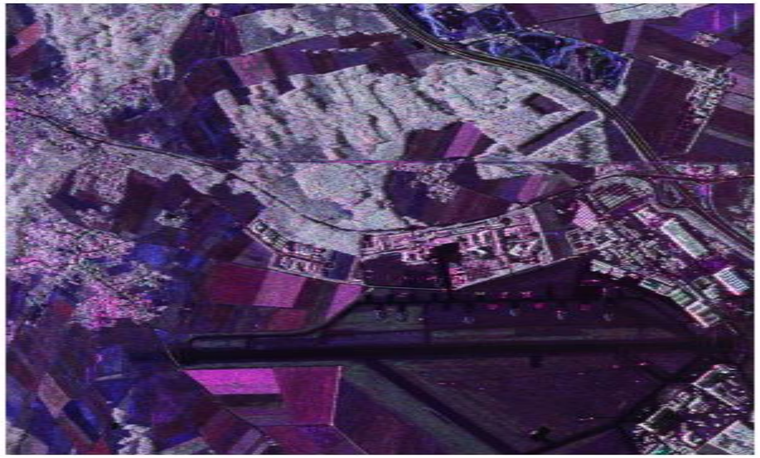


out-of-distribution? → **quantum** **physics** to rescue

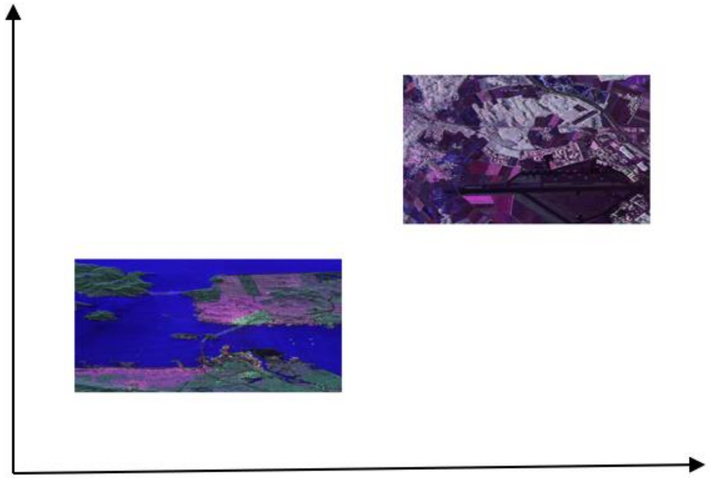
Quantum Variational Algorithm for Earth Observation: Case II



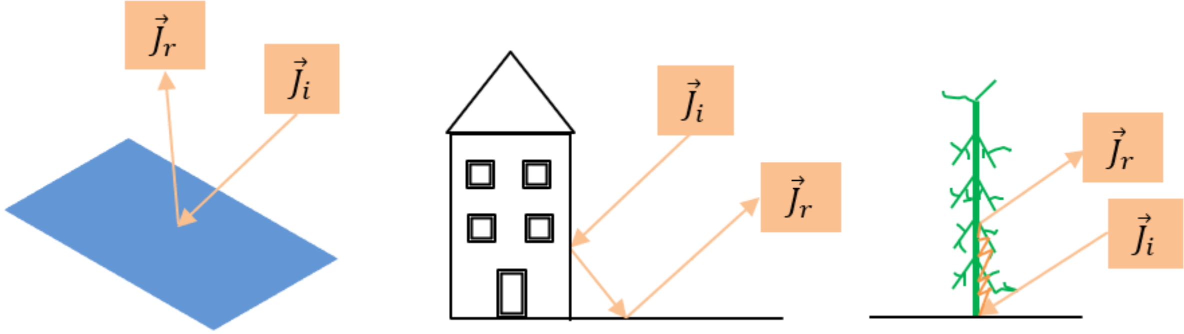
PolSAR: San Francisco



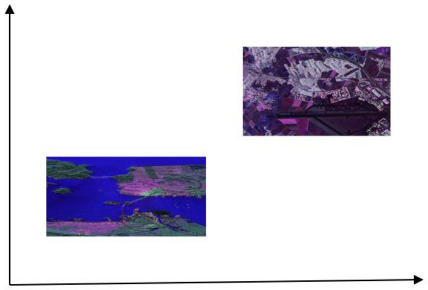
PolSAR: DLR Oberpfaffenhofen



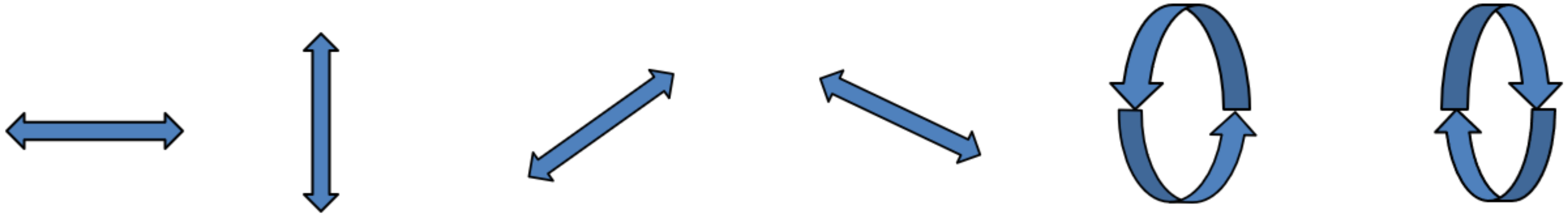
Quantum Variational Algorithm for Earth Observation: Case II



(a) (b) (c)

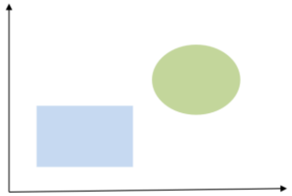


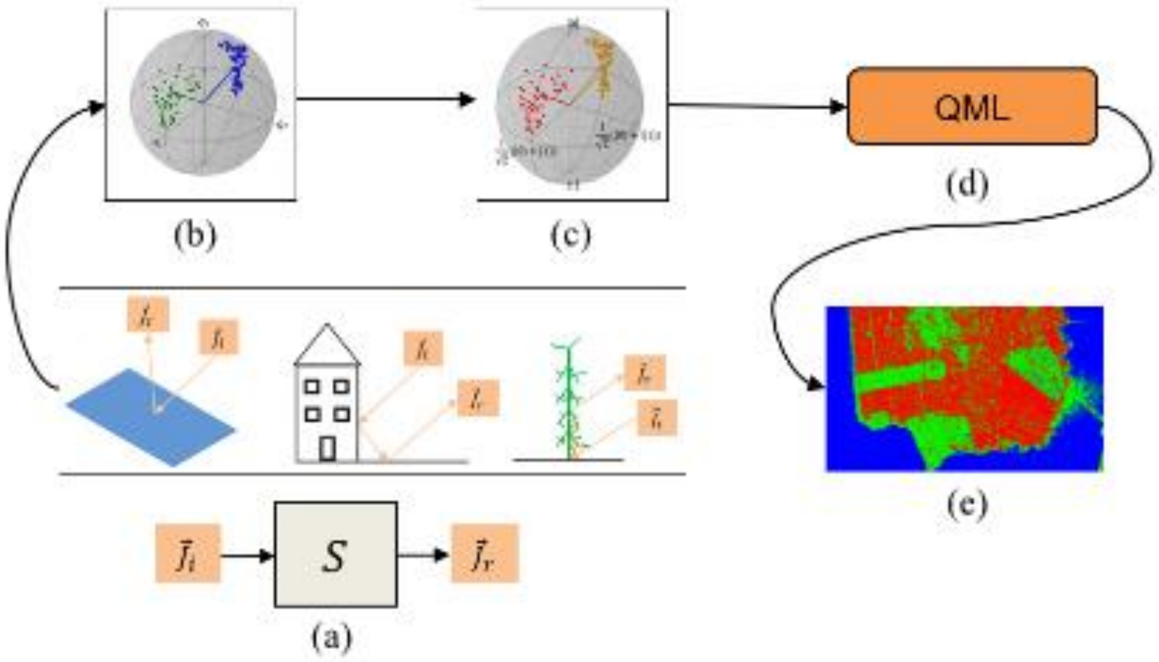
Say one 0-10 cm, other 20-30 cm wavelength

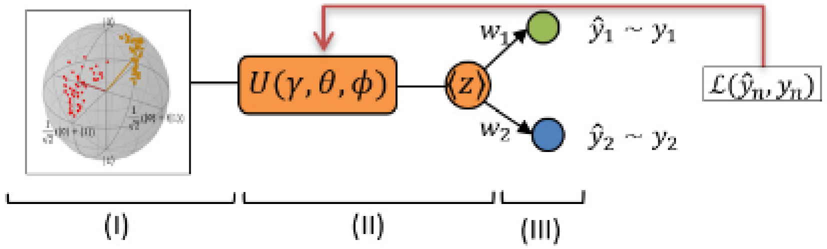


$|0\rangle$ $|1\rangle$ $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ $\frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$ $\frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)$

Quantum Variational Algorithm for Earth Observation: Case II

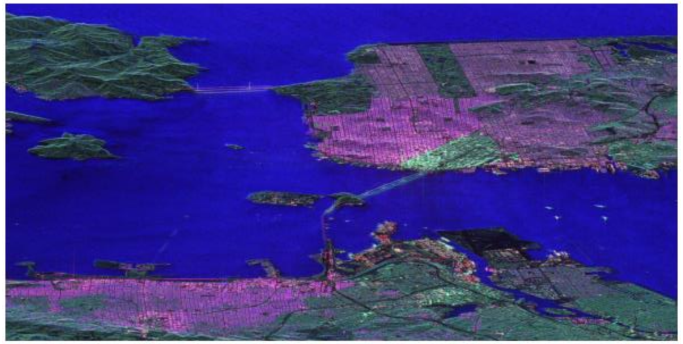




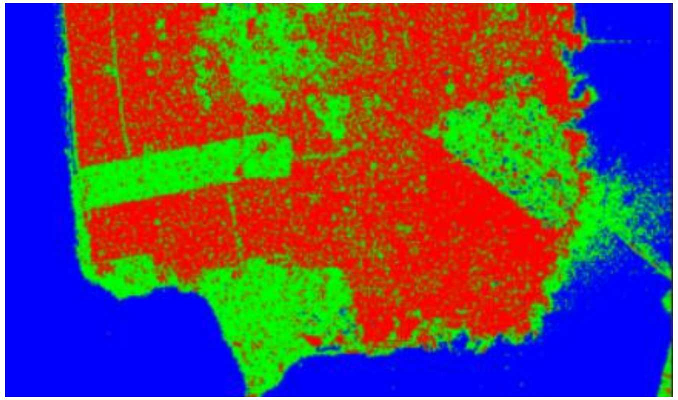


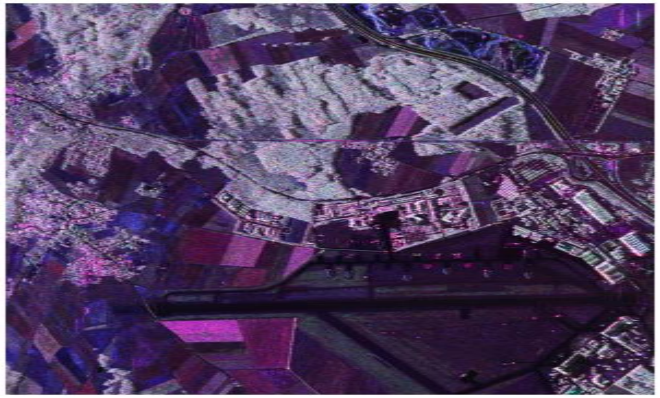
S. Otgonbaatar and M. Datcu, "Natural Embedding of the Stokes Parameters of Polarimetric Synthetic Aperture Radar Images in a Gate-Based Quantum Computer," in *IEEE Transactions on Geoscience and Remote Sensing*, doi: 10.1109/TGRS.2021.3110056.

Quantum Variational Algorithm for Earth Observation: Case II

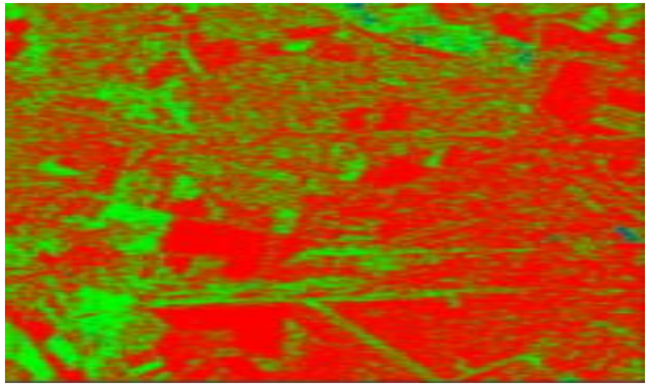


Trained

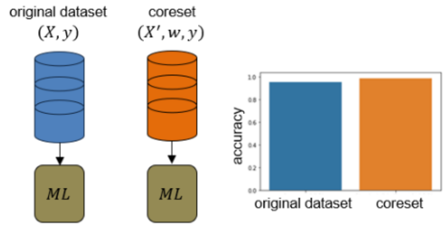


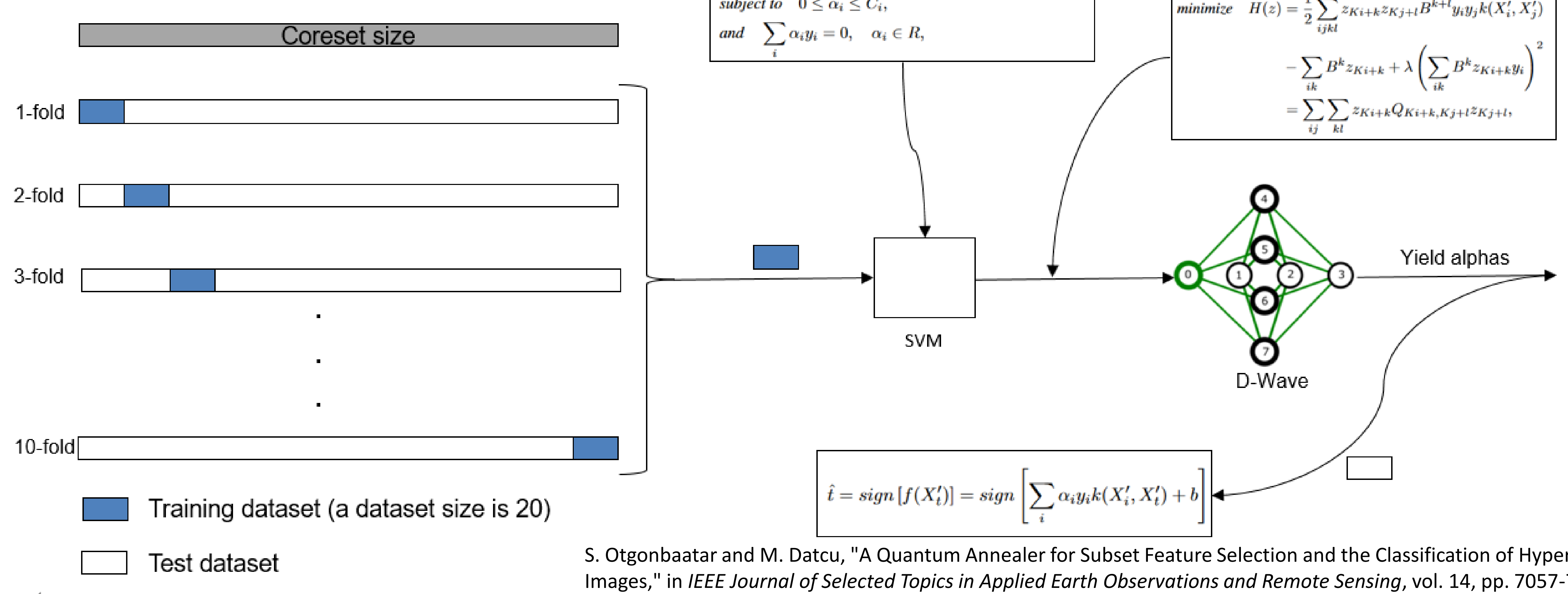


Tested



Quantum Annealer for Earth Observation: Case III





weighted cSVM

$$\text{minimize} \quad H(\alpha) = \frac{1}{2}\sum_{ij}\alpha_i\alpha_j y_i y_j k(X'_i, X'_j) - \sum_i \alpha_i$$
$$\text{subject to} \quad 0 \le \alpha_i \le C_i,$$
$$\text{and} \quad \sum_i \alpha_i y_i = 0, \quad \alpha_i \in R,$$

weighted cSVM to QUBO

$$\text{minimize} \quad H(z) = \frac{1}{2}\sum_{ijkl} z_{Ki+k} z_{Kj+l} B^{k+l} y_i y_j k(X'_i, X'_j) - \sum_{ik} B^k z_{Ki+k} + \lambda\left(\sum_{ik} B^k z_{Ki+k} y_i\right)^2 = \sum_{ij}\sum_{kl} z_{Ki+k} Q_{Ki+k,Kj+l} z_{Kj+l},$$

$$\hat{t} = sign\,[f(X'_t)] = sign\left[\sum_i \alpha_i y_i k(X'_i, X'_t) + b\right]$$

S. Otgonbaatar and M. Datcu, "A Quantum Annealer for Subset Feature Selection and the Classification of Hyperspectral Images," in *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 7057-7065, 2021, doi: 10.1109/JSTARS.2021.3095377.

Quantum Annealer for Earth Observation: Case III

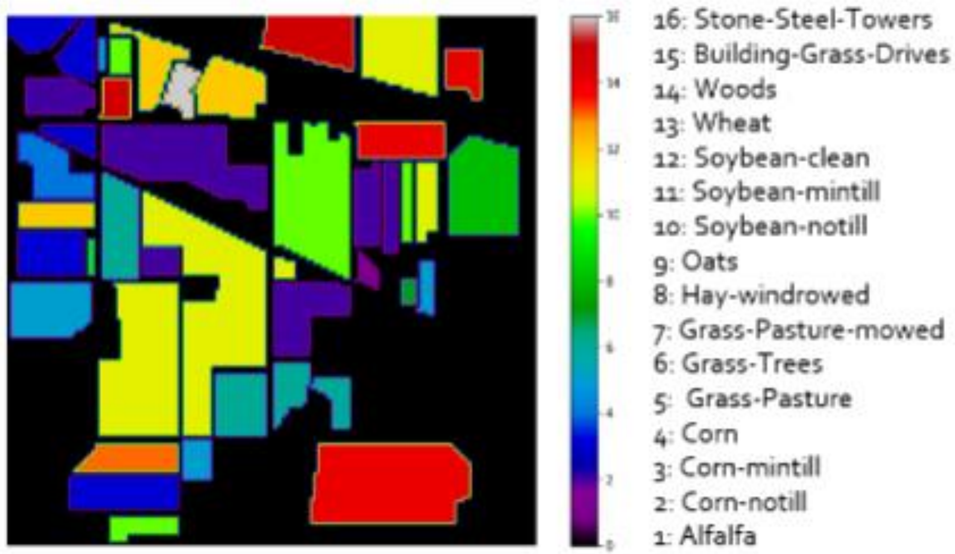
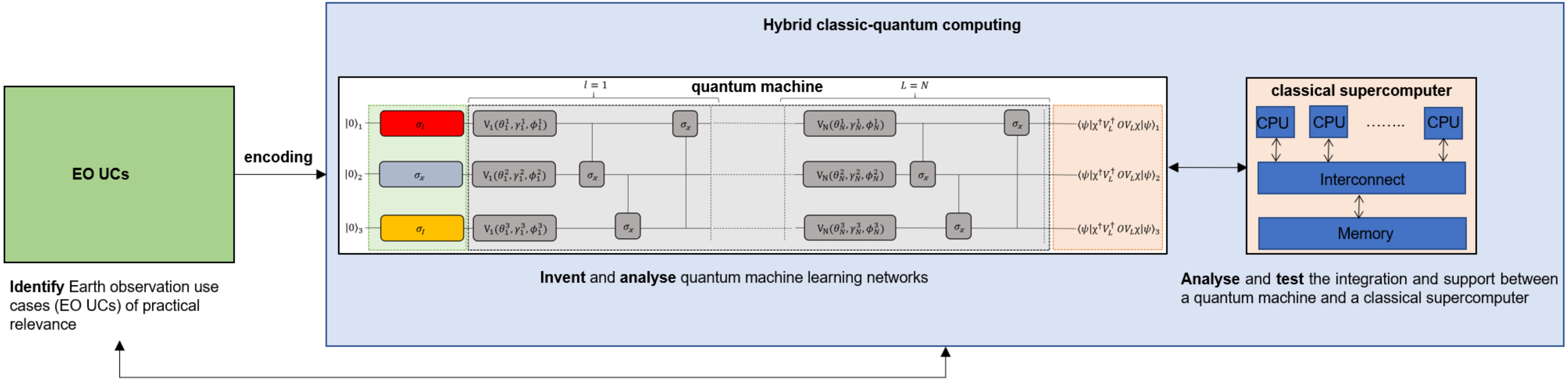


Fig. 2. Our Indian Pine HSI with 16 classes: {1: Alfalfa, 2: Corn-notill, 3: Corn-mintill, 4: Corn, 5: Grass-Pasture, 6: Grass-Trees, 7: Grass-Pasture-mowed, 8: Hay-windrowed, 9: Oats, 10: Soybean-notill, 11: Soybean-mintill, 12: Soybean-clean, 13: Wheat, 14: Woods, 15: Building-Grass-Drives, 16: Stones-Steel-Towers.

Classes	Data Size	Coreset Size	KL Divergence
$\{-1, +1\}$	100	20	0.008194
{setosa, versicolour}	100	22	0.053002
{1, 2}	295	79	0.573451
{2, 3}	452	56	0.003121
{3, 4}	214	33	0.000600
{4, 5}	144	41	0.017201
{5, 6}	243	41	0.001823
{6, 7}	758	125	0.492636
{urban area, sea water}	61,465	501	0.125072
{vegetation, sea water}	61,465	343	0.272749

The Last Slide of This Talk

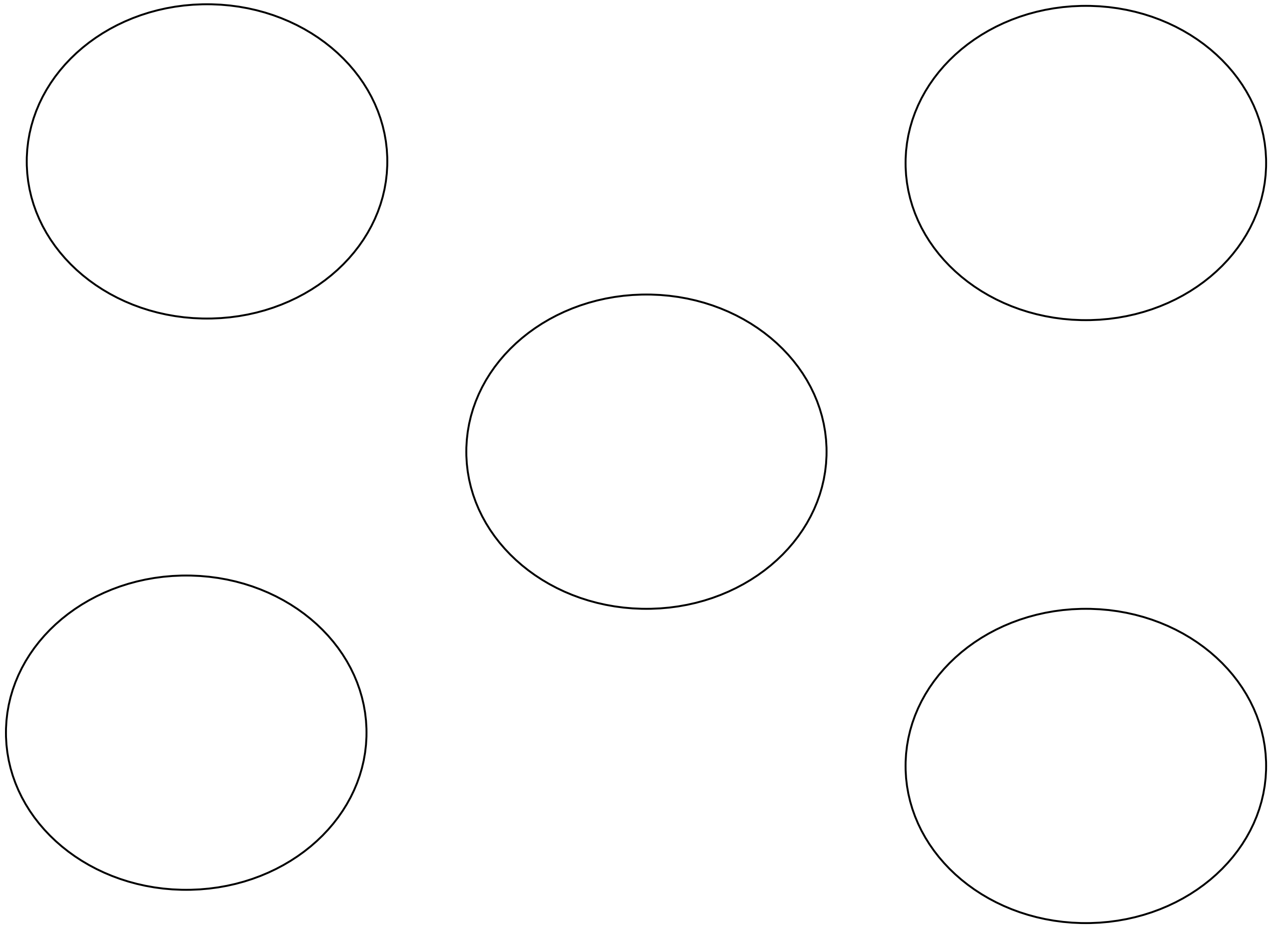
Input big data → How to → Small quantum computer



Analyse a quantum machine and a classical supercomputer for real-world, large scale datasets in order to obtain quantum advantage as early and efficiently as possible

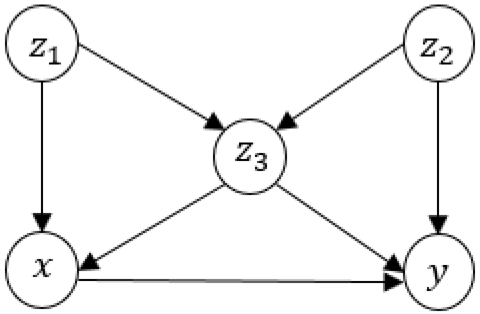
Next Question: Can we really demonstrate quantum advantage by leveraging a HPC+nQC system over a conventional heterogeneous system, since we now know what is a quantum computer and how to make it work for large data sets?

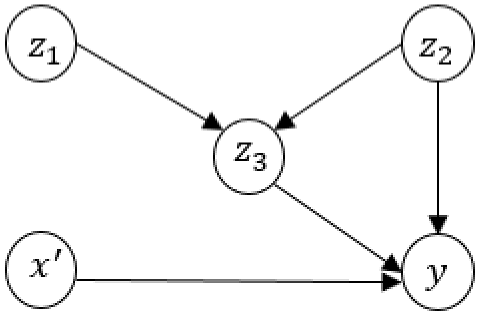
Next Question: quantum advantage on HPC+nQC system



Soronzonbold Otgonbaatar et al IGARSS 2022

Next Question: quantum advantage on HPC+nQC system





N	Number of causal DAGs
0	1
1	1
2	3
3	25
4	543
5	29281
6	3781503
7	1138779265
8	783702329343
9	1213442454842881
10	4175098976430598143
11	31603459396418917607425
12	521939651343829405020504063
13	18676600744432035186664816926721
14	1439428141044398334941790719839535103

Soronzonbold Otgonbaatar et al IGARSS 2022