© April 2023| IJIRT | Volume 9 Issue 11 | ISSN: 2349-6002

A Study and Observation of Quantum Techniques for Particle Swarm Optimization (PSO)

Prateek Karkera[1], Srivaramangai. R[2]
[1]*Department of Information Technology, University of Mumbai*
[2]*Department of Information Technology, University of Mumbai*

***Abstract*—The recent advances in real Quantum Computing have lent credibility and acclaim to the idea of using Parameterized Quantum Computing methods as hypotheses for Quantum-Classical Hybrid Machine Learning Systems. Quantum-Classical Hybrid systems are the next step towards comprehensive Quantum Enhanced Systems. They have already shown great promise and potential in solving supervised and generative learning tasks with recent works demonstrating their superiority in specialized Artificial Intelligence tasks as well. However, the largest impact that Quantum Advantage can bring about in present-day systems lies in optimizing the hardest and most complex parallel learning algorithms. From this perspective, this research compares three of the most challenging artificial intelligence algorithms that illustrate the leverage which can be obtained by harnessing the properties of quantum computing. In this paper, Quantum Enhanced Reinforcement Learning, Genetic Algorithms and Particle Swarm Optimization are explored with an emphasis on the applications of Particle Swarm Optimization.**

***Index Terms*—Quantum Computing, Particle Swarm Optimization, Machine Learning, Variational Quantum Algorithm, Quantum Reinforcement Learning, Quantum Genetic Algorithm.**

I. INTRODUCTION

In the 1980s, quantum computing emerged as a field of study when researchers began to investigate computational models that incorporated principles from quantum mechanics [1]. Benioff and Deutsch, who worked on the concept of quantum Turing machines and universal quantum computation [2, 3], were among the pioneers in this field. Subsequent research has focused on the application of quantum computing to the simulation of quantum systems [4-6]. The advancement of the discipline, however, was spurred by Peter Shor's 1994 discovery of an effective quantum method for determining the prime factors of composite integers, which exposed the weaknesses of conventional cryptography protocols [7]. Since then, the research of quantum algorithms has expanded to cover a variety of applications, such as search and optimization, machine learning, simulation of quantum systems, and cryptography [8]. Quantum computing has been a rapidly developing field for the past four decades, with many disciplines contributing to the study and implementation of quantum algorithms. Quantum computers are unique tools that offer significant computational power, particularly in fields with high computational demands. To successfully implement quantum algorithms, the smallest units of quantum information, called qubits, must be as reliable as classical bits. However, they must also be protected from noise that causes decoherence and be controllable by external agents. This control includes the ability to create entanglement between qubits and to perform measurement operations to extract the output of quantum computation. Quantum computing is a type of computing that uses quantum mechanics, a branch of physics, to store and process information. It has the potential to perform certain types of computation much faster than classical computers, which use bits to store and process information. In classical computers, a bit is a unit of information that can be either a 0 or a 1. Quantum computers use quantum bits, or qubits, which can represent both a 0 and a 1 at the same time. This property, known as superposition, allows quantum computers to perform certain calculations much more quickly than classical computers. A 2^n-dimensional complex Hilbert space, or $H = (C2)^{\otimes n}$, can be used to represent a quantum system made up of n qubits. A vector $|\psi\rangle \in H$ of unit norm $\langle\psi|\psi\rangle = 1$ is used to express the quantum state of the object, and the bra-ket notation has been used to describe vectors $|\psi\rangle$, their conjugate transpose $\langle\psi|$, and inner products $\langle\psi|\psi'\rangle$ in H. $|0\rangle = (1, 0)^T$ and $|1\rangle = (0, 1)^T$ represent single-qubit computational basis states, and their tensor products describe generic computational basis states, such as |10⟩ = |1⟩ ⊗ |0⟩ = (0, 0, 1, 0). A very intriguing use of upcoming quantum computers is hybrid quantum machine learning models, which generate a hypothesis family for a learning task using parametrized and data-dependent quantum computations and train them using a conventional optimization technique [9, 10]. These models, which include parametrized quantum circuits (PQCs) [11] as examples, have shown promise in classification [12–16], generative modelling [18–19], and clustering [20] problems. Additionally, PQCs have shown learning advantages in tasks that were artificially created under the principles of complexity theory [21–24].

.

II. QUANTUM TECHNIQUES IN PSO

A. Variational Quantum Algorithms (VQA)

A key tactic to deal with the limitations such as the limited numbers of qubits and noise processes that limit circuit depth is the use of variational quantum algorithms (VQAs), which train a parameterized quantum circuit using a classical optimizer. VQAs appear to be the best option for getting quantum advantage because they have already been proposed for nearly all applications that researchers have thought of for quantum computers.

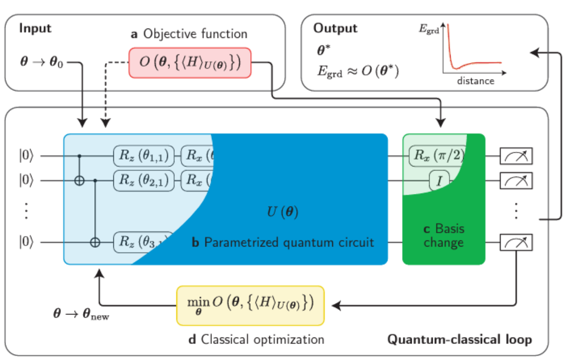

Fig.1 Diagrammatic Representation of a Variational Quantum Algorithm (VQA). [10]

The Fig. 1 describes the diagrammatic representation of a Variational Quantum Algorithm (VQA). A VQA workflow can be divided into four main components: a) the objective function O that encodes the problem to be solved; b) the parameterized quantum circuit (PQC) U, which variables θ are tuned to minimise the objective; c) the measurement scheme, which performs the basis changes and measurements needed to compute expectation values that are used to evaluate the objective; and d) the classical optimizer that minimizes the objective. The PQC can be defined heuristically, following hardware-inspired ansätze, or designed from the knowledge about the problem Hamiltonian H. Inputs of a VQA are the circuit ansatz U(θ) and the initial parameter values θ0. Outputs include optimized parameter values θ∗ and the minimum of the objective.

B. Quantum Reinforcement Learning (QRL)

Hybrid quantum machine learning models that combine classical optimization algorithms with quantum computations have shown promising application for near-term quantum computers [9, 10]. These models use parameterized and data-dependent quantum computations to define a hypothesis family for a specific learning task. For example, parametrized quantum circuits (PQCs) [11] have been successful in solving classification [12-16], generative modelling [18, 19] and clustering [20] problems, and have demonstrated learning advantages in artificially constructed tasks [14, 21], some of which are based on complexity-theoretic assumptions [21-24]. Reinforcement learning (RL) is a field that could greatly benefit from the use of a powerful hypothesis family, as seen with the improvement in learning performance provided by deep neural networks (DNNs) in RL [25]. At the core of RL algorithms is a PQC that takes the agent's state in the environment (e.g., a NumPy array) as input and outputs a vector of expectation values. These expectation values are then processed to create the agent's policy or approximate Q-values. In this way, PQCs serve a similar role to deep neural networks in modern deep RL algorithms. One common way to encode an input vector in a PQC is through the use of single-qubit rotations, where the rotation angles are controlled by the components of the input vector. To create a highly expressive model, these single-qubit encodings are not performed just once, but are "re-uploaded" multiple times and interlaid with variational gates. The layout of such a PQC is depicted in Fig.2.

A way to further enhance the expressivity and trainability of data re-uploading PQCs is to use trainable input-scaling parameters λ for each encoding gate of the PQC, and trainable observable weights w at its output.

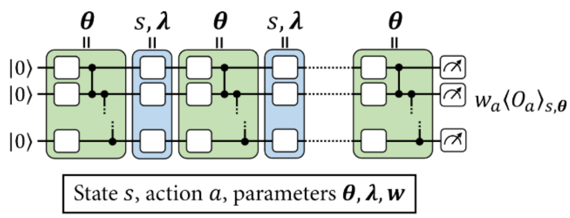

Fig.2 Layout of a Parameterized Quantum Circuit interlayed with variational gates. [17]

C. Quantum Genetic Algorithms (QGA)

Quantum genetic algorithms (QGAs) were first introduced in 1996 by Narayanan and Moore, and have been used to solve the Travelling Salesman Problem (TSP) with great success [26]. QGAs, which are based on the principles of genetic algorithms, have demonstrated significant advantages over traditional genetic algorithms in terms of efficiency, speed of convergence, global optimization capability, and robustness, even with small populations [27, 28]. In QGAs, genetic code is represented using a quantum state vector, and evolution of the chromosome is achieved through the use of quantum logic gates. However, there are still some limitations to conventional QGAs that require further exploration. The three parts of the genetic algorithm need to be specified and then suitable inputs to be selected in order to show a proof-of-concept for a quantum-assisted genetic algorithm.

Recombination: To generate two new individuals, this recombination procedure takes two individuals, randomly chooses a connected cluster of spins in which the individuals differ, then flips this cluster in each individual.

Mutation: To mutate a state S, a reverse anneal is performed which is initialised at state S. This reverse anneal applies a transverse field to evolve the classical starting state into a quantum superposition of states, then removes the transverse field to settle on a new classical state.

Selection: For selection, truncation selection is used; i.e., simply the best N individuals at the end of each generation is kept. This has the advantage of simplicity, but it can lead to loss of population diversity. Quantum-assisted genetic algorithm can be described by the following pseudocode:

population = N random states
FOR generation = 1 TO num_generations DO
  mutate each individual in population with
  probability mutation_rate
  add the mutated states to population
  randomly match the individuals to make
  recombination_rate × |population| pairs
  recombine each pair using an isoenergetic cluster
  move to make 2 new offspring
  add the offspring to population
BREAKIF the algorithm has reached a stopping criterion,
  (e.g., by timing out or reaching a certain energy)
  discard individuals from population to maintain
  the desired population size of N
END FOR

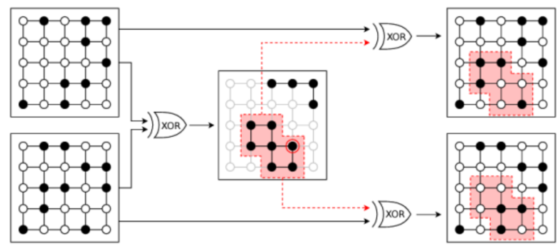

Fig.3 Example of recombination on a 2D lattice. [29]

Fig.3 gives an example of recombination on a 2D lattice via isoenergetic cluster move. The two "parent" individuals (left) are chosen and their symmetric difference (xor for binary states) is considered (middle). From the symmetric difference, a variable in which the two inputs differ is selected uniformly at random (circled) and its connected component (dashed and shaded region) specifies the cluster of variables to flip. To obtain the child individuals (right), the variables in this cluster are flipped in each of the parent individuals.

D. Quantum Particle Swarm Optimization (QPSO)

In 1995, researchers began working on ways to improve the performance of the particle swarm optimization (PSO) algorithm [30, 31]. However, Van den Bergh [32] demonstrated that PSO is not a global optimization algorithm. To address this issue, Sun et al. [33] combined quantum theory with PSO to create the quantum-behaved particle swarm optimization (QPSO) algorithm. The Quantum Particle Swarm Optimization (QPSO) algorithm is an optimization algorithm that uses a population of particles to search for the global minimum or maximum of an objective function. The algorithm is inspired by the behaviour of swarms of birds or bees, where each individual particle is guided by its own experience and the collective experience of the group. The PSO is based on the concept of dividing the population of particles into teams and allowing the teams to evolve and compete with each other. Each particle belongs to a team, and the global best position of each team is used to update the velocities and positions of the particles in the team. A velocity updating function updates the velocity of the particle based on a combination of the personal best position and the global best position. The personal best position is the best position that the particle has encountered so far, and the global best position is the best position that has been found by any particle in the population. The position updating function is then used to update the position of the particle based on the updated velocity. This allows the particle to move in the direction and at the magnitude determined by the velocity. These functions are typically used in combination with a stopping criterion, which determines when the algorithm should terminate. The stopping criteria may be based on the number of iterations, the time taken, or the quality of the solutions found. Once the stopping criteria is met, the algorithm returns the global best position as the final solution to the optimization problem. The velocities and positions of the particles in the Quantum Particle Swarm Optimization (QPSO) algorithm are adjusted based on a combination of their personal best positions, the global best position, and their current velocities and positions. This allows the particles to move towards the most promising areas of the search space and to converge towards the optimal solution to the optimization problem. The velocity and position of a particle can be calculated from the following equation.

(i) **v[i][j]** = w * v[i][j] + c1 * r1 * (p[i][j] - x[i][j]) + c2 * r2 * (g[j] - x[i][j]) + c3 * r3 * (g[j] - x[i][j])

(ii) **x[i][j]** = x[i][j] + v[i][j]

In this update rule, v[i][j] and x[i][j] represent the velocity and position of the i-th particle at the j-th dimension, respectively. w is the inertia weight, which determines the influence of the previous velocity on the current velocity. p[i][j] is the personal best position of the i-th particle at the j-th dimension. g[j] is the global best position at the j-th dimension. c1, c2, and c3 are constants that determine the influence of the personal best position, the global best position, and the teamwork evolutionary strategy on the velocity, respectively. r1, r2, and r3 are random weights that are used to add a degree of randomness to the update rule. The c3 term in the update rule represents the influence of the teamwork evolutionary strategy on the velocity. In the Quantum Particle Swarm Optimization (QPSO) algorithm, each particle calculates its attraction point as a weighted average of its own historical optimal position and the global best position of the group. However, this calculation has two drawbacks:

(i) Each particle's position depends on the historical optimal position of the group, in addition to its own learning experience. This leads to a rapid decline in diversity in large groups, reducing the algorithm's ability to solve complex multi-peak optimization problems.

(ii) The distribution space of each particle's attraction point decreases during the evolution of the algorithm. The particles are confined to a rectangle defined by vertices p[i][j] and g[i][j]. As the algorithm progresses, the function approaches the global best. This also means that the algorithm may become trapped in a local optimum in the final stages.

The development of the Quantum Particle Swarm Optimization (QPSO) algorithm by Sun et al. in [33] aimed to improve the performance of the Particle Swarm Optimization (PSO) algorithm by combining it with quantum theory. QPSO has been shown to be effective in finding global optimal solutions in the search space, and has been demonstrated to improve the standard PSO algorithm on various benchmark functions. While QPSO is able to find the global optimal solution in the case of infinite searching iterations, this is not realistic in practical problems, as optimization algorithms are only allowed a limited number of iterations to find the optimal solution.

III. OBSERVATIONS OF QT FOR PSO

In the coming years, the integration of quantum computing into artificial intelligence is expected to drive rapid development in related research areas like machine learning. This is because quantum computing offers the potential for faster and more effective algorithms. For example, Zhaokai et al. [50] implemented a quantum support vector machine algorithm for an optical character recognition problem using a 4-qubit processor and the NMR technique with 13C-iodotrifluoroethylene and a spectrometer at 306 K. Their results showed the potential for quantum computing to improve the performance of machine learning algorithms. Looking ahead, researchers are already working towards the physical realization of quantum computers with 50-100 qubits, as demonstrated by Veldhorst et al., [51] which will provide the necessary hardware to build a quantum computer. As these technologies continue to advance, even more impressive developments are expected to be seen in the field of artificial intelligence and machine learning. For instance, the study by Farhi et al. [12] showed that higher resolution input and a more powerful model, Classical (Full) model, made classification problems easier for the CNN. Interestingly, a classical model with similar power (~32 parameters), Classical (Fair) model, was also able to achieve a similar accuracy albeit in a fraction of the time. This result highlights the importance of exploring and optimizing the trade-offs between computational resources and model complexity in the design of machine learning algorithms.

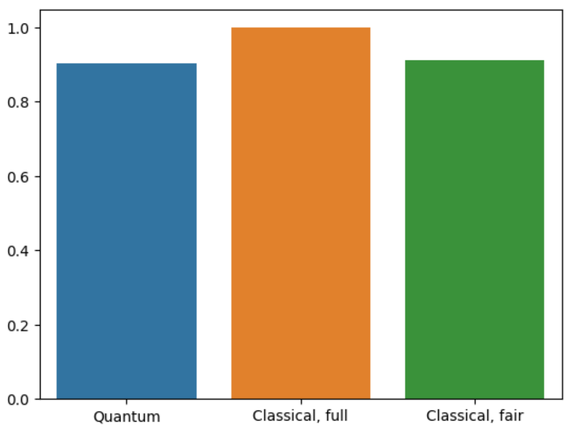

Fig.4 Accuracy Comparison of Quantum, Full Classical, and Fair Classical Neural Networks. [12]

Fig.4 depicts a comparison of accuracy among a Quantum Neural Network, a Full Classical Neural Network, and a Fair Classical Neural Net with similar power (~32 parameters) to that of the Quantum Neural Network. The scale denotes the level of precision with which the models were able to classify the given data. The closer the value is to 1.0, the higher the accuracy of the model.

Benchmarking environments from OpenAI Gym (Greg Brockman et al., 2019) [34], for which good and simple DNN policies are known, were considered in a numerical investigation. In these environments, it was demonstrated that PQC policies could achieve comparable performance. Inspired by the classification task studied by Vojtech Havlicek et al. [14], which was conjectured to be classically hard by the authors, analogous RL environments were constructed. In these environments, an empirical learning advantage of PQC policies over standard DNN policies used in deep RL was shown. Additionally, RL environments with a provable gap in performance between a family of PQC policies and any efficient classical learner were constructed. These environments were based on the work of Yunchao Liu et al. [23] and involved the embedding of the discrete logarithm problem (DLP) into a learning setting. The DLP, which can be solved by Shor's celebrated quantum algorithm (Peter W. Shor, [35]), is widely believed to be classically hard to solve (Manuel Blum and Silvio Micali, [36]). Recently, several works have explored hybrid quantum approaches for reinforcement learning (RL). Chen et al. [37] and Lockwood and Si [38] trained PQC-based agents in classical RL environments using a value-based approach, in which PQCs were used as value-function approximators instead of direct policies. These works tested their learning agents on OpenAI Gym environments, but did not achieve sufficiently good performance according to the Gym specifications. Skolik et al. [39] showed that, using some of the design choices for PQCs in RL described in our work, such as data re-uploading circuits with trainable observable weights and input scaling parameters, a value-based approach can be used to solve these environments. Wu et al. [41] introduced an actor-critic approach to quantum RL, using both a PQC actor (or policy) and a PQC critic (or value-function approximator). These were trained in quantum environments that provide a quantum state to the agent, which then responds with a continuous classical action, making it a very different learning setting to ours. Jerbi et al. [42] also described a hybrid quantum-classical algorithm for value-based RL, but used energy-based neural networks (e.g., deep and quantum Boltzmann machines) as function-approximation models rather than PQCs. This research presents an alternative approach to leveraging quantum effects in the design of QRL agents compared to earlier approaches such as those by Dong et al. [43], Paparo et al. [44], Dunjko et al. [45], Crawford et al. [46], and Neukart et al. [47], which are mainly based on Grover's search algorithm or quantum annealers to speed up sampling routines. Quantum computers, such as IBM's quantum processor, are currently available for experimentation through cloud computing platforms. However, even though quantum evolutionary algorithms (QGAs) are inspired by quantum computing principles, they are usually performed on classical computers. Researchers believe that this will change once QGAs are designed and implemented on quantum computers. This would allow for faster research on higher-order QGAs (Nowotniak, R. et al., [53]), QGAs with entanglement (Choy, C.K. et al., [54]), and hybrid QGAs with quantum optimization algorithms (Duan, H.B. et al., [55]). To make this transition, it is important to train future computer scientists in the fundamentals of quantum mechanics as mentioned by Mermin, N.D., in the 2003 study. The implementation of quantum algorithms will lead to a significant increase in the use of quantum evolutionary algorithms in various fields, including the analysis of cancer microarray data (Sardana, M. et al., [57]) and classical engineering optimization problems (Mani, A. and Patvardhan, C., [58]), as well as in artificial intelligence (Draa, A. et al., [59]) and artificial life (Alvarez-Rodriguez, U. et al., [60]). Lahoz-Beltra in the 2008 study explained how in the future, quantum computing may also revolutionize our understanding of Darwinism. In the meantime, current software and hardware technologies can be utilized, such as the NVIDIA CUDA platform and the Matlab GPU library (Montiel, O. et al., [62]), to design more efficient QGAs. Florian Neukart et al., 2018 developed QGAs based on Grover's search algorithm (Lov K Grover, [48]) or quantum annealers (Mark W Johnson et al., [49]) to improve sampling routines. Searching for the optimal subset of features is a difficult optimization problem, according to research by Blum and Langley in 1997 [63]. To solve this problem, various methods have been developed, which can be divided into classical and metaheuristic methods, as stated by Cotta and Moscato in 2003 [64]. In their study, the authors have focused on metaheuristic methods, which use a powerful mechanism to find better solutions, in contrast to heuristics which simply suggest solutions without ensuring that they are the best ones, as shown by Zhao et al. in 2013 [65]. There are many metaheuristic algorithms used in feature selection, such as genetic algorithms (GA), clonal selection algorithm (CSA), ant colony optimization (ACO), and particle swarm optimization (PSO). PSO is a relatively new metaheuristic, but it has been demonstrated to be simple and effective compared to other methods, according to Zhao et al. [65]. Despite its effectiveness, it is important to continue improving this method.

V. CONCLUSION

In this paper the various quantum techniques for optimization were studied and it has been observed that a lot of potential is there for the improvement and use of quantum swam particle optimization. Since it is a niche area, the experimental research needs to be done in implementing the techniques to actual optimization problems. More improvements are needed in the existing techniques too. The authors have taken this step ahead for development of a methodology for the implementation which can be discussed further once the performance analysis is done and the results are reported.

REFERENCES

[1] Kaiser, David. How the hippies saved physics: science, counterculture, and the quantum revival. WW Norton & Company, 2011.

[2] Benioff, Paul. "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines." Journal of statistical physics 22, no. 5 (1980): 563-591.

[3] Deutsch, David. "Quantum theory, the Church–Turing principle and the universal quantum computer." Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences 400, no. 1818 (1985): 97-117.

[4] Feynman, Richard P. "Quantum mechanical computers." Found. Phys. 16, no. 6 (1986): 507-532.

[5] Lloyd, Seth. "Universal quantum simulators." Science 273, no. 5278 (1996): 1073-1078.

[6] Manin, Yuri. "Computable and uncomputable." Sovetskoye Radio, Moscow 128 (1980).

[7] Shor, Peter W. "Algorithms for quantum computation: discrete logarithms and factoring." In Proceedings 35th annual symposium on foundations of computer science, pp. 124-134. Ieee, 1994.

[8] Montanaro, Ashley. "Quantum algorithms: an overview." npj Quantum Information 2, no. 1 (2016): 1-8.

[9] Preskill, John. "Quantum computing in the NISQ era and beyond." Quantum 2 (2018): 79.

[10] Bharti, Kishor, Alba Cervera-Lierta, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote et al. "Noisy intermediate-

scale quantum (NISQ) algorithms." arXiv preprint arXiv:2101.08448 (2021).
[11] Benedetti, Marcello, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. "Parameterized quantum circuits as machine learning models." Quantum Science and Technology 4, no. 4 (2019): 043001.
[12] Farhi, Edward, and Hartmut Neven. "Classification with quantum neural networks on near term processors." arXiv preprint arXiv:1802.06002 (2018).
[13] Schuld, Maria, Alex Bocharov, Krysta M. Svore, and Nathan Wiebe. "Circuit-centric quantum classifiers." Physical Review A 101, no. 3 (2020): 032308.
[14] Havlíček, Vojtěch, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. "Supervised learning with quantum-enhanced feature spaces." Nature 567, no. 7747 (2019): 209-212.
[15] Schuld, Maria, and Nathan Killoran. "Quantum machine learning in feature Hilbert spaces." Physical review letters 122, no. 4 (2019): 040504.
Peters, Evan, João Caldeira, Alan Ho, Stefan Leichenauer, Masoud Mohseni, Hartmut Neven, Panagiotis Spentzouris, Doug Strain, and Gabriel N. Perdue. "Machine learning of high dimensional data on a noisy quantum processor." npj Quantum Information 7, no. 1 (2021): 1-5.
[16] Peters, Evan, João Caldeira, Alan Ho, Stefan Leichenauer, Masoud Mohseni, Hartmut Neven, Panagiotis Spentzouris, Doug Strain, and Gabriel N. Perdue. "Machine learning of high dimensional data on a noisy quantum processor." npj Quantum Information 7, no. 1 (2021): 1-5.
[17] Jerbi, Sofiene, Casper Gyurik, Simon Marshall, Hans Briegel, and Vedran Dunjko. "Parametrized quantum policies for reinforcement learning." Advances in Neural Information Processing Systems 34 (2021): 28362-28375.
[18] Liu, Jin-Guo, and Lei Wang. "Differentiable learning of quantum circuit born machines." Physical Review A 98, no. 6 (2018): 062324.
[19] Zhu, Daiwei, Norbert M. Linke, Marcello Benedetti, Kevin A. Landsman, Nhung H. Nguyen, C. Huerta Alderete, Alejandro Perdomo-Ortiz et al. "Training of quantum circuits on a hybrid quantum computer." Science advances 5, no. 10 (2019): eaaw9918.
[20] Otterbach, Johannes S., Riccardo Manenti, Nasser Alidoust, A. Bestwick, M. Block, B. Bloom, S. Caldwell et al. "Unsupervised machine learning on a hybrid quantum computer." arXiv preprint arXiv:1712.05771 (2017).
[21] Huang, Hsin-Yuan, Michael Broughton, Masoud Mohseni, Ryan Babbush, Sergio Boixo, Hartmut Neven, and Jarrod R. McClean. "Power of data in quantum machine learning." Nature communications 12, no. 1 (2021): 1-9.
[22] Du, Yuxuan, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao. "Expressive power of parametrized quantum circuits." Physical Review Research 2, no. 3 (2020): 033125.
[23] Liu, Yunchao, Srinivasan Arunachalam, and Kristan Temme. "A rigorous and robust quantum speed-up in supervised machine learning." Nature Physics 17, no. 9 (2021): 1013-1017.
[24] Sweke, Ryan, Jean-Pierre Seifert, Dominik Hangleiter, and Jens Eisert. "On the quantum versus classical learnability of discrete distributions." Quantum 5 (2021): 417.
[25] Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves et al. "Human-level control through deep reinforcement learning." nature 518, no. 7540 (2015): 529-533.
[26] Narayanan, Ajit, and Mark Moore. "Quantum-inspired genetic algorithms." In Proceedings of IEEE international conference on evolutionary computation, pp. 61-66. IEEE, 1996.
[27] Van Meter, Rod, Kae Nemoto, and W. Munro. "Communication links for distributed quantum computation." IEEE Transactions on Computers 56, no. 12 (2007): 1643-1653.
[28] Zhou, Shu, Wei Pan, Bin Luo, Wei-Li Zhang, and Ying Ding. "A novel quantum genetic algorithm based on particle swarm optimization method and its application." ACTA ELECTONICA SINICA 34, no. 5 (2006): 897.
[29] King, James, Masoud Mohseni, William Bernoudy, Alexandre Fréchette, Hossein Sadeghi, Sergei V. Isakov, Hartmut Neven, and Mohammad H. Amin. "Quantum-assisted genetic algorithm." arXiv preprint arXiv:1907.00707 (2019).
[30] Ratnaweera, Asanga, Saman K. Halgamuge, and Harry C. Watson. "Self-organizing hierarchical particle swarm optimizer with time-varying

acceleration coefficients." IEEE Transactions on evolutionary computation 8, no. 3 (2004): 240-255.
[31] Messerschmidt, Leon, and Andries P. Engelbrecht. "Learning to play games using a PSO-based competitive learning approach." IEEE Transactions on Evolutionary Computation 8, no. 3 (2004): 280-288.
[32] Van Den Bergh, Frans. "An analysis of particle swarm optimizers." PhD diss., University of Pretoria, 2007.
[33] Sun, Jun, C-H. Lai, Wenbo Xu, Yanrui Ding, and Zhilei Chai. "A modified quantum-behaved particle swarm optimization." In International Conference on Computational Science, pp. 294-301. Springer, Berlin, Heidelberg, 2007.
[34] Brockman, Greg, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. "Openai gym." arXiv preprint arXiv:1606.01540 (2016).
[35] Shor, Peter W. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer." SIAM review 41, no. 2 (1999): 303-332.
[36] Blum, Manuel, and Silvio Micali. "How to generate cryptographically strong sequences of pseudo random bits." In Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali, pp. 227-240. 2019.
[37] Chen, Samuel Yen-Chi, Chao-Han Huck Yang, Jun Qi, Pin-Yu Chen, Xiaoli Ma, and Hsi-Sheng Goan. "Variational quantum circuits for deep reinforcement learning." IEEE Access 8 (2020): 141007-141024.
[38] Lockwood, Owen, and Mei Si. "Reinforcement learning with quantum variational circuit." In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, vol. 16, no. 1, pp. 245-251. 2020.
[39] Skolik, Andrea, Sofiene Jerbi, and Vedran Dunjko. "Quantum agents in the gym: a variational quantum algorithm for deep q-learning." Quantum 6 (2022): 720.
[40] Pérez-Salinas, Adrián, Alba Cervera-Lierta, Elies Gil-Fuster, and José I. Latorre. "Data re-uploading for a universal quantum classifier." Quantum 4 (2020): 226.
[41] Wu, Shaojun, Shan Jin, Dingding Wen, and Xiaoting Wang. "Quantum reinforcement learning in continuous action space." arXiv preprint arXiv:2012.10711 (2020).
[42] Jerbi, Sofiene, Lea M. Trenkwalder, Hendrik Poulsen Nautrup, Hans J. Briegel, and Vedran Dunjko. "Quantum enhancements for deep reinforcement learning in large spaces." PRX Quantum 2, no. 1 (2021): 010328.
[43] Dong, Daoyi, Chunlin Chen, Hanxiong Li, and Tzyh-Jong Tarn. "Quantum reinforcement learning." IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 38, no. 5 (2008): 1207-1220.
[44] Paparo, Giuseppe Davide, Vedran Dunjko, Adi Makmal, Miguel Angel Martin-Delgado, and Hans J. Briegel. "Quantum speedup for active learning agents." Physical Review X 4, no. 3 (2014): 031002.
[45] Dunjko, Vedran, Jacob M. Taylor, and Hans J. Briegel. "Quantum-enhanced machine learning." Physical review letters 117, no. 13 (2016): 130501.
[46] Crawford, Daniel, Anna Levit, Navid Ghadermarzy, Jaspreet S. Oberoi, and Pooya Ronagh. "Reinforcement learning using quantum Boltzmann machines." arXiv preprint arXiv:1612.05695 (2016).
[47] Neukart, Florian, David Von Dollen, Christian Seidel, and Gabriele Compostella. "Quantum-enhanced reinforcement learning for finite-episode games with discrete state spaces." Frontiers in physics 5 (2018): 71.
[48] Grover, Lov K. "A fast quantum mechanical algorithm for database search." In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, pp. 212-219. 1996.
[49] Johnson, Mark W., Mohammad HS Amin, Suzanne Gildert, Trevor Lanting, Firas Hamze, Neil Dickson, Richard Harris et al. "Quantum annealing with manufactured spins." Nature 473, no. 7346 (2011): 194-198.
[50] Zhaokai, Li, Liu Xiaomei, and Xu Nanyang. "Experimental realization of quantum artificial intelligence." arXiv preprint arXiv:1410.1054 (2014).
[51] Veldhorst, Menno, C. H. Yang, J. C. C. Hwang, W. Huang, J. P. Dehollain, J. T. Muhonen, S. Simmons et al. "A two-qubit logic gate in silicon." Nature 526, no. 7573 (2015): 410-414.
[52] Vu, C. "IBM makes quantum computing available on IBM cloud to accelerate innovation." IBM News Room (2016).
[53] Nowotniak, Robert, and Jacek Kucharski. "Higher-order quantum-inspired genetic algorithms." In 2014 Federated Conference on Computer Science and Information Systems, pp. 465-470. IEEE, 2014.

[54] Choy, Chee Ken, Kien Quang Nguyen, and Ruck Thawonmas. "Quantum-inspired genetic algorithm with two search supportive schemes and artificial entanglement." In 2014 IEEE Symposium on Foundations of Computational Intelligence (FOCI), pp. 17-23. IEEE, 2014.
[55] Duan, Hai-bin, Chun-fang Xu, and Zhi-Hui Xing. "A hybrid artificial bee colony optimization and quantum evolutionary algorithm for continuous optimization problems." International Journal of Neural Systems 20, no. 01 (2010): 39-50.
[56] Mermin, N. David. "From Cbits to Qbits: Teaching computer scientists quantum mechanics." American Journal of Physics 71, no. 1 (2003): 23-30.
[57] Sardana, Manju, R. K. Agrawal, and Baljeet Kaur. "Clustering in conjunction with quantum genetic algorithm for relevant genes selection for cancer microarray data." In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 428-439. Springer, Berlin, Heidelberg, 2013.
[58] Mani, Ashish, and C. Patvardhan. "An adaptive quantum evolutionary algorithm for engineering optimization problems." International Journal of Computer Applications 1, no. 22 (2010): 43-48.
[59] Draa, Amer, Souham Meshoul, Hichem Talbi, and Mohamed Batouche. "A quantum-inspired differential evolution algorithm for solving the N-queens problem." neural networks 1, no. 2 (2011): 21-27.
[60] Alvarez-Rodriguez, Unai, Mikel Sanz, Lucas Lamata, and Enrique Solano. "Artificial life in quantum technologies." Scientific Reports 6, no. 1 (2016): 1-9.
[61] Lahoz-Beltra, Rafael. "Quantum genetic algorithms for computer scientists." Computers 5, no. 4 (2016): 24.
[62] Montiel, Oscar, Ajelet Rivera, and Roberto Sepúlveda. "Design and acceleration of a quantum genetic algorithm through the matlab GPU library." In Design of intelligent systems based on fuzzy logic, neural networks and nature-inspired optimization, pp. 333-345. Springer, Cham, 2015.
[63] Blum, Avrim L., and Pat Langley. "Selection of relevant features and examples in machine learning." Artificial intelligence 97, no. 1-2 (1997): 245-271.
[64] Cotta, Carlos, and Pablo Moscato. "The k-Feature Set problem is W [2]-complete." Journal of Computer and System Sciences 67, no. 4 (2003): 686-690.
[65] Zhao, Jing, Jun Sun, and Wenbo Xu. "A binary quantum-behaved particle swarm optimization algorithm with cooperative approach." International Journal of Computer Science Issues (IJCSI) 10, no. 1 (2013): 112.
[66] De Castro, Leandro N., and Fernando J. Von Zuben. "Learning and optimization using the clonal selection principle." IEEE transactions on evolutionary computation 6, no. 3 (2002): 239-251.
[67] Dorigo, Marco, Mauro Birattari, and Thomas Stutzle. "Ant colony optimization." IEEE computational intelligence magazine 1, no. 4 (2006): 28-39.
[68] Kennedy, James, and Russell Eberhart. "Particle swarm optimization." In Proceedings of ICNN'95-international conference on neural networks, vol. 4, pp. 1942-1948. IEEE, 1995.