A Novel Quantum Machine Learning Algorithm Based on Kronecker Reed-Muller Forms

Bryan Lee

bryanlee99@gmail.com | bryanlee@college.harvard.edu

Portland State University

Abstract

Machine learning is being broadly applied in many areas to process large quantities of data, establish prediction models and predict the complex system behavior. In today's ever-growing big data era, the total store of digital data already amounts to 487 billion gigabytes and this figure is expected to double in size every 18 months. In fact, it would take a classical computer enormous amount of time to manipulate big data and recommend the desired behavior for complex systems in machine learning. In contrast, quantum machine learning can accelerate the learning process significantly with its power of parallel quantum computing, especially when handling exponentially growing data sets for machine learning.

In my research, a novel quantum machine learning algorithm was invented to execute complex machine learning predictive analysis where many variables, patterns and test models are possible. The machine learning problem in this research is modeled as finding the simplest Kronecker Reed-Muller (KRO) forms for incomplete Boolean functions with multiple "don't know" values. An innovative implementation of Grover's search algorithm was developed to enable quadratic accelerations for machine learning. A MATLAB simulator was developed to simulate quantum computer calculations to find the exact minimum KRO expression which yields the simplified machine learning prediction function compared against FPRM (Fixed Polarity Reed-Muller) forms. This research also integrated the simulator with a HR-OS1 humanoid robot to validate functionality. Over 65,000 MATLAB simulation runs were completed to validate KRO form minimization results and 600 machine learning simulation runs were tested, spanning different design permutations for variety of input variables. The machine learning prediction function complexity was reduced by up to ~33% in experiments. The HR-OS1 humanoid robot and MATLAB simulator integration validated the correctness of the algorithm.

Personal Section

For many children, robots are captivating due to the incredible feats they can accomplish, such as navigating rugged terrain and competing in sports such as soccer. I was no different. Since I was young, my enjoyment for building and programming robots has allowed me to apply my passion for creativity in design to goal-oriented engineering. This passion eventually led to an iterative process that gradually brought a once imagined product to life. As I grew older and began participating in competitive robotics, such as through FIRST Tech Challenge, I realized that behind each robot's action was an algorithm directing its decision-making process. Among these algorithms, those that stood out most to me were in machine learning, a type of artificial intelligence which allows computers to dynamically react to new situations without being explicitly programmed. After discovering these algorithms, I was soon engaged in the field as I learned that machine learning algorithms had beneficial applications in nearly every computational area, ranging from medical diagnosis to email filtering. My interest in machine learning was further strengthened by my love for mathematics, which is a basis for many machine learning algorithms: with my background in competitive math, I had always enjoyed the intricate explanations and novel creativity required by solutions to problems in mathematics which eventually drew me towards research relating to the field.

During an evening quantum computing seminar at Portland State University, Professor Marek Perkowski discussed the relatively new area of quantum machine learning. As the lecture continued, I gradually realized that I had come across a field with an ideal intersection between my passions for mathematics and machine learning. Following this seminar, I started reading published works in the field which introduced quantum analogs of algorithms ranging from support vector machines to k-nearest neighbors. Afterwards, I combined my previous knowledge of various Boolean function types and machine learning to create my current project. This project, which I have conducted under the guidance of Professor Perkowski, applies quantum computing's accelerated computation abilities to increase the training speed of machine learning algorithms, which can often be slow when faced with large quantities of data.

In order to complete my project, I read textbooks which discussed fundamental theories in quantum computing to establish a foundation in the field and studied academic papers related to my research in Grover's Search Algorithm, a quantum algorithm which accelerates search speed in an unsorted dataset for a given element. In addition, I independently studied linear algebra and differential equations in order to understand the mathematics behind the quantum algorithms. Although establishing a foundation in a new topic area, such as quantum computing, required a significant amount of time dedication and perseverance despite initially obscure explanations, conducting research in this area has been an extremely rewarding

experience as it has introduced me to new methods of thought through theories that defy our intuition and has exposed me to new areas of science and mathematics.

For aspiring high school researchers, my most important piece of advice would be to not limit yourself to working on projects you feel comfortable with at first as many initially daunting or uncertain areas of research likely have invaluable knowledge and significant potential to make a positive impact on you as a researcher and on society. For me, my research project required me to explore difficult areas of mathematics and science. However, due to my research, I was able to gain knowledge in these areas and contribute a novel quantum machine learning algorithm to a field which could significantly improve the way computation is conducted in the future.

1. Introduction

Machine learning has broad applications and can give computers the ability to learn without being explicitly programmed [1]. Supervised machine learning techniques are among major machine learning methods to infer a prediction function from labeled training data which consist of a set of training examples where each example is a pair of an input vector of features and an output decision. After the learning, the machine learning test phase uses test data to validate the prediction function. One challenge in supervised machine learning is accelerating the learning as it can be very time-consuming and in many cases is a bottleneck of machine learning processes when handling big data sets, complex use cases and input patterns. For example, Le, et al. [2] indicated it requires ~3 days to use model parallelism on a cluster with 1,000 machines to process 10 million 200 x 200 pixel images for deep learning in feature recognition. In addition, machine learning acceleration is essential for the learning process. Abbeel [3] reported the critical need for helicopter maneuvering to learn and make decision in less than 2 seconds.

Quantum machine learning is an emerging interdisciplinary research area, combining quantum physics and computer science to apply quantum mechanics to methods of machine learning. Quantum machine learning uses the parallelism of quantum computation for machine learning acceleration (elaborated upon in Section 2.3) [4]. Researchers reported machine learning classification problems can be formulated as a generic quantum computing problem to find the target function that represents all potential solutions for machine learning classification. For example, machine learning using Support Vector Machines (SVMs) can be implemented using the combination of several known quantum algorithms for classification (Rebentrost, et al.[5]; S. Yoo, et al. [6]; Cai, et al.[7]; Lloyd, et al [8]).

My proposed quantum machine learning algorithm is a novel approach which converts the learning samples into an incompletely specified Boolean function for classification problems. An incompletely specified Boolean function is a function with multiple "don't know" values (values which are not covered by the training data) along with values "0" or "1" corresponding to negative and positive decisions, respectively. A complex system can have multiple incompletely specified Boolean functions to handle multiple inputs and outputs. A new Kronecker Reed-Muller (KRO) spectral transform, to convert the Boolean function into a KRO form, is used to minimize such functions (i.e. with as many zero spectral coefficients as possible). Spectral coefficients are defined as the constants corresponding to each product term in a KRO form. There are a total of 3^n KRO forms (informally proven in Section 2.2) and the proposed algorithm is able to identify the exact minimum expression out of all such forms which cover each "don't know" for a function of n input variables. In previous research, a related Fixed Polarity Reed-Muller (FPRM) form based quantum approaches (Li et al. [9], Sarabi and Perkowski [10]) was studied; however,

given that the 3^n KRO forms are a superset of the 2^n FPRM forms (informally proven in Section 2.2), a KRO-based quantum algorithm can produce a solution of either better or the same complexity than that of FPRM forms. In addition, this algorithm, based on repeated application of ternary/binary Grover's quantum search, provides quadratic acceleration compared to an equivalent classical search algorithm. To demonstrate functionality, the quantum machine learning algorithm is applied to the humanoid HR-OS1 robot system [12] with a Microsoft 6CH-00001 Webcam, which captures digital images (e.g., object color, object shape, people faces, etc.) and feeds into quantum machine learning software. The software processes digital images using MATLAB and outputs recommendation for a set of complex Boolean functions for intelligent robot decisions.

The quantum machine learning algorithm and quantum circuit design from this research established a new quantum machine learning branch. This research can be applied to accelerate any machine learning that has complex binary inputs and outputs including intelligent robot decision making, medical diagnosis, cybersecurity, business, and industrial planning when handling big data inputs and complex use cases.

2. Experiments

2.1. Research System Architecture Design

The experimental system is composed of Intel Core i5 2.30 GHz Laptop, a Raspberry Pi 2 powered HR-OS1 robot system, 6CH-00001 digital camera to capture images and MATLAB based quantum machine learning and computer vision software that I wrote. This set-up is shown in Figure 1.

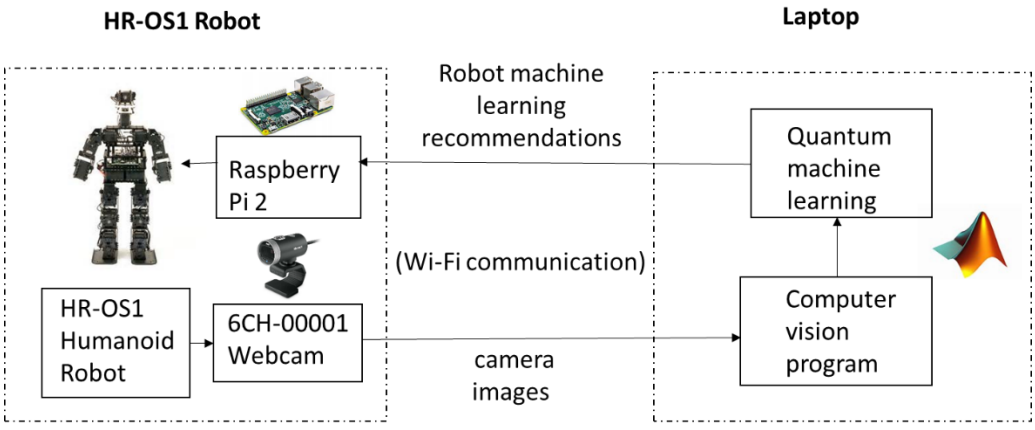


Figure 1: System setup illustration- HR-OS1 with webcam communicates with laptop via Wi-Fi. The laptop has the MATLAB based quantum machine learning algorithm and computer vision programs. The Raspberry Pi 2 on the HR-OS1 receives predictions and then makes intelligent robot movements.

2.2 Experiment Methods

Boolean functions are used in many applications when the inputs and outputs are binary. In fact, multiple Boolean functions can be combined to handle systems with multiple binary output variables. A n-variable Boolean function is as following:

$$f\ (x_1, x_2, \dots\dots, x_n) = \sum_{i=0}^{2^n-1} m_{\mathrm{i}} \hat{x}_1 \hat{x}_2 \dots\dots \hat{x}_{\mathrm{n}} \tag{1}$$

Where $m_{\mathrm{i}} \in \{0,1\}$ are referred to as minterm coefficients of function *f*. $\hat{x}_i$ represents a literal in either positive polarity (x_i) or negative polarity ($\bar{x}_i$). Note that m_{i} corresponds to combinations of the binary positives and negations for $\hat{x}_1 \hat{x}_2 \dots\dots \hat{x}_{\mathrm{n}}$. For example, m_1 corresponds to $\overline{x_1}\ \overline{x_2}\ \overline{x_3}\ \dots \overline{x_{\mathrm{n-1}}}\ x_{\mathrm{n}}$ (000…001).

Because all base binary functions are disjoint, the OR operators in (1) can be changed to EXOR (Exclusive OR) operators using Reed-Muller expansions. In order to generalize Reed-Muller expansions, Davio [13] introduced following three fundamental expansions:

$$f(x_1, x_2, \dots, x_n) = 1 \cdot f_0(x_2, \dots, x_n) \oplus x_1 f_2(x_2, \dots, x_n) \text{ In short } f = f_0 \oplus x_1 f_2 \tag{2}$$

$$f(x_1, x_2, \dots, x_n) = 1 \cdot f_1(x_2, \dots, x_n) \oplus \overline{x_1} f_2(x_2, \dots, x_n) \text{ In short } f = f_1 \oplus \overline{x_1} f_2 \tag{3}$$

$$f(x_1, x_2, \dots, x_n) = \overline{x_1} f_0(x_2, \dots, x_n) \oplus x_1 f_1(x_2, \dots, x_n) \text{ In short } f = \overline{x_1} f_0 \oplus x_1 f_1 \tag{4}$$

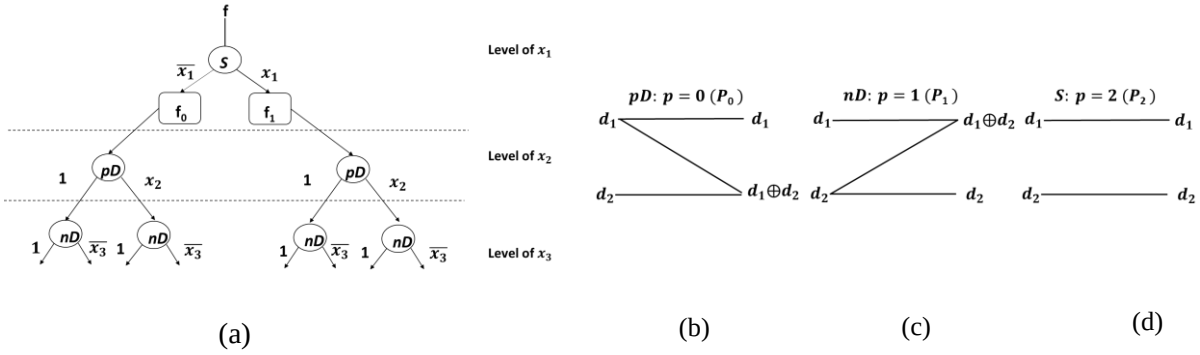
Where f_0 is f with x_1 replaced by 0 (negative cofactor of variable x_1), f_1 is f with x_1 replaced by 1 (positive cofactor of variable x_1), and $f_2 = f_0 \oplus f_1$, which is called the Boolean Difference. $\oplus$ represents the EXOR operation. Eq. (2) is a Positive Davio (pD) Expansion, Eq. (3) is a Negative Davio (nD) Expansion, Eq. (4) is a Shannon Expansion (S).

When only pD and nD are applied, an FPRM form is obtained for n inputs, which leads to 2^n possible FPRM forms, as there are two possible expansions for each input variable. When S, pD and nD are all applied to all possible n variable combinations, 3^n possible KRO forms exist for similar reasons. Every expansion is a collection of product terms joined by the EXOR operator. As an example, FPRM forms of three variable functions can be expressed as follows:

$$a_0 1 \oplus a_1 \hat{x}_1 \oplus a_2 \hat{x}_2 \oplus a_3 \hat{x}_3 \oplus a_{12}\ \hat{x}_1 \hat{x}_2 \oplus a_{13} \hat{x}_1 \hat{x}_3 \oplus a_{23}\ \hat{x}_2 \hat{x}_3 \oplus a_{123} \hat{x}_1 \hat{x}_2 \hat{x}_3 \tag{5}$$

where $a_i \in \{0,1\}$ are binary spectral coefficients and $\hat{x}_i$ represents a literal in either positive polarity (x_1, x_2, x_3) or negative polarity ($\overline{x_1}, \overline{x_2}, \overline{x_3}$), where positive and negative polarity correspond to variables which are complemented or not complemented, respectively. Note that the FPRM form has variables of constant polarity while KRO expansions allow differing polarities for the same variable.

Machine learning in this project is formulated as finding the minimal KRO form and is motivated by Occam's Razor Principle [14]: "*Among competing hypotheses, the one with the fewest assumptions should be selected*". This is done by minimizing the number of non-zero a_i coefficients for KRO forms. A KRO transformation matrix can be created by recursively applying the Kronecker product (see Figure 2.a). Figures 2.b, 2.c and 2.d show "butterfly" transformations which realize the KRO form.



(a) (b) (c) (d)

Figure 2: KRO expansions: (a) Illustration of the recursive KRO expansion for variables x_1, x_2 and x_3; (b), (c) and (d) are butterfly schematic diagrams for pD, nD and S expansions, respectively. d_1 and d_2 on the left of the butterflies are inputs; the output of the expansions is on the right-hand side of the butterflies.

In such a butterfly, edges represent multiplicative weights and vertices represent additions (in this case of modulo-2 [U. Kalay et al. [14]). This transformation is known as the KRO transform and is commonly characterized by a linear transformation matrix. The structure of this transformation matrix can be expressed as a Kronecker (or tensor) matrix product where each dependent variable is represented by a kernel matrix for a given polarity. As an example, the transformation matrix for Fixed Polarity Reed Muller (FPRM) transformation is given:

$$M_{P1} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \tag{6}$$

$$M_{N1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \tag{7}$$

Where (6) is for the noncompleted variables and (7) is for the complemented variables. In addition to M_{P1} and M_{N1}, Shannon expansion matrix M_{S1} is as following:

$$M_{S1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{8}$$

Figure 3 shows a three variable (x_1, x_2, x_3) example of a KRO expansion using a butterfly flow diagram for selected polarities (x_1: Shannon; x_2: Positive Davio; and x_3: Negative Davio).

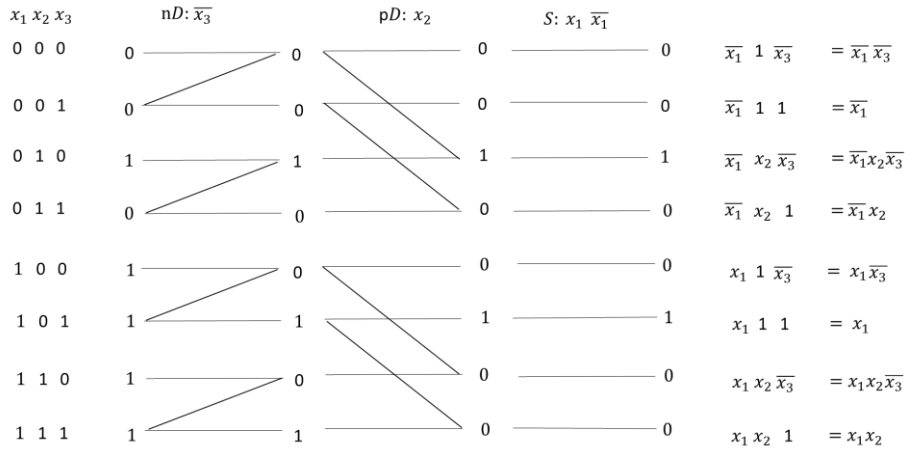


Figure 3: Example for KRO expansions butterfly flow diagram with 3 variables (x_1, x_2, x_3). The left-hand side has all input minterms and their values; and the right-hand side has the spectral coefficient and literals.

In classical computing, the butterfly circuit for one polarity is constructed and the cost (number of ones in the spectrum) is calculated. Then, the butterfly circuit for next polarity is constructed and its cost were counted. As this is repeated for all polarities, this method would be very slow. My research achieves a quadratic speedup due to quantum computing's computational parallelism based on a superposition of states, which will be explained in session 2.3.

Figure 4 shows the truth table for the machine learning formulation used in this research. The output is the machine learning hypothesis (prediction) for all input variables. "T" represents machine learning predictions and the hypothesis is expected to predict "T" values. In Figure 4.b and 4.c, Karnaugh Map uses an array of squares where each represents a different combination of input variables and their output values. KRO maps to rectangles that cover Karnaugh Map. Minimizing KRO forms means minimizing the number of rectangles.

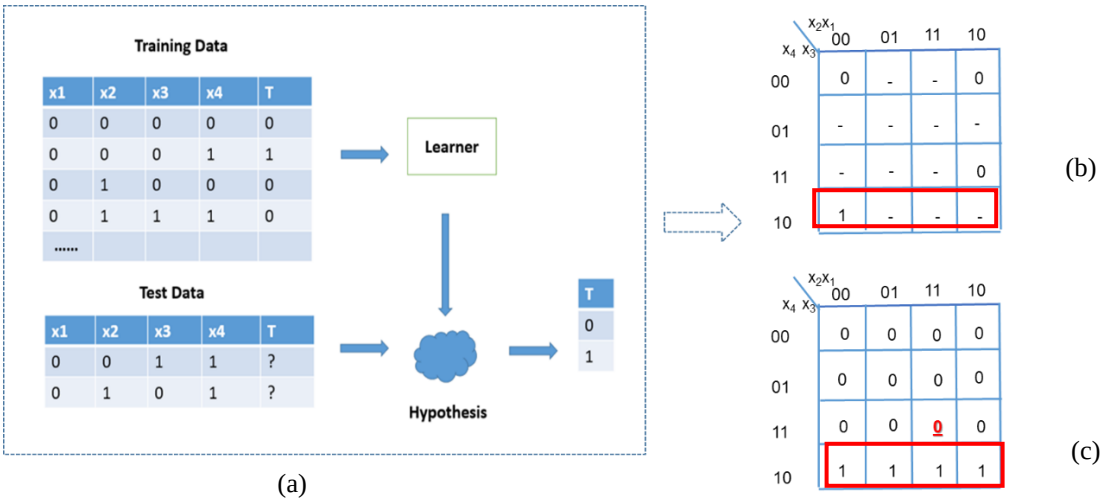
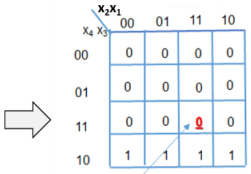


Figure 4: Quantum machine learning example illustration- (a) Training data have 4 training data sets. The output "T" is predicted after the learner completes the learning and provides the hypothesis of prediction function. (b) Karnaugh Map for training data sets. The '-' indicates target values with unknown. Specifically, the combination of input variables was not seen in the training data set. (c) The predicted "T" values from the hypothesis are shown in Karnaugh Map with "1" or "0", an absence or presence of output, which replace "-".

In this research, I developed a MATLAB simulator to simulate the Quantum machine learning algorithm. In addition, I also developed the program to integrate with the humanoid HR-OR1 robot in order to validate the correctness of the algorithm. The MATLAB based computer vision image processing program to extract the features from the matrix of pixels for face recognition. The publicly used database of faces from AT&T and Cambridge University was utilized for face recognition validations. There are 40 peoples with 10 pictures taken at different time and stored in the database.

After image processing, Boolean variables to indicate feature detections (e.g.,x_1, x_2, x_3, x_4) were fed into the quantum machine learning program, which completes the learning and predicts robot actions. Figure 5 shows an example of the image processing and prediction function inputs and respective outputs.

Input Variables	Input Images	Image Process Method	Image Process Outcome	Input to Machine Learning
X1 (Blue Circle)		HSV (color)	(Blue, Circle) Yes	1
x2 (Red Square)		HSV (color)	(Red, Square) Yes	1
X3 (Face Image for Person A)	Face Image A from Webcam	HOG (Features)	Matched classifier A in Database	1
X4 (Face Image for Person B)	Face Image B from Webcam	HOG (Features)	Matched classifier B in Database	1



Karnaugh Map from Figure 2 example for prediction: 0 target value when x1, x2, x3, x4=1

Figure 5: The robot camera loads the image into MATLAB program through Wi-Fi communications. The MATLAB processed the images (e.g., face matched the pre-defined classifier in the database) and then input the x_1, x_2, x_3 , and x_4 data values into the MATLAB machine learning program. The MATLAB program can predict "T" (Machine learning predictions) and send recommendations to control robot movements. Multiple machine learning Boolean functions are needed to control complex robot actions.

The case study from this research is to show that the quantum machine learning algorithm and computer vision program can extract features and make predictions for real time learning. The MATLAB simulator that I developed outputs minimum KRO form, KRO expansion polarity and machine learning run time. This machine learning simulator is generic and can be applied to any Boolean function applications.

2.3. System Design and Implementation

2.3.1 Novel Quantum Machine Learning Algorithm with Grover's Search Algorithm

The quantum computation model breaks down the quantum system into two components: quantum bits (qubits), particles which make up the quantum system, and quantum gates, transforms applied to the qubits. The qubits undergo a quantum phenomenon called "superposition" where they can exist in many states simultaneously and store an exponentially large amount of information. This effect can be employed to provide significant speed-up in computation [17]. In addition to qubits $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$, $(|\alpha|^2 + |\beta|^2 = 1)$ with base states represented by kets $|0\rangle = \begin{bmatrix}1\\0\end{bmatrix}$ and $|1\rangle = \begin{bmatrix}0\\1\end{bmatrix}$, this research also uses qutrits (ternary quantum trits) where Dirac notation for the state is:

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle = \alpha\begin{bmatrix}1\\0\\0\end{bmatrix} + \beta\begin{bmatrix}0\\1\\0\end{bmatrix} + \gamma\begin{bmatrix}0\\0\\1\end{bmatrix} \quad (9)$$

where $|\alpha|^2 + |\beta|^2 + |\gamma|^2 = 1$. The α, β, and γ components are referred to as amplitudes and their respective squares $|\alpha|^2$, $|\beta|^2$ and $|\gamma|^2$, are the probability amplitudes of each state. At any moment, a register of n-qutrits will be in a composite superposition state of different base qutrit kets. The notation for representing the state of a n-qubit or n-qutrit quantum system is as following:

$$|\Psi\rangle = |\Psi_1 \dots \Psi_n\rangle = |\Psi_1\rangle \otimes \dots \otimes |\Psi_n\rangle \tag{10}$$

Where $\otimes$ represents Kronecker product operation. The resulting state space, $|\Psi\rangle$, is a $2^n \times 1$ vector for n-qubit or $3^n \times 1$ vector for n-qutrit state space.

A quantum gate (e.g., Hadamard gate, Toffoli gate) is an operation which is applied to the qubit or qutrits. The quantum gates can be represented by unitary matrices and are used to realize quantum circuit logic operations. The most common quantum gate is the Hadamard gate. The Hadamard transform can be applied to ket $|0\rangle = \begin{bmatrix}1\\0\end{bmatrix}$, generating the uniform superposition as inputs to quantum circuits shown below:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix}1 & 1\\1 & -1\end{bmatrix} \tag{11}$$

I formulated a general approach to reduce logic minimization problems into the problem of building a multi-valued quantum oracle to determine if the sum of KRO non-zero coefficients is less than a threshold value. For every input combination, the quantum oracle answers only yes/no at its output. A quantum oracle, shown in Figure 9, is a quantum circuit that creates a mapping to vectors of its input qubits (additional information regarding the oracle's implementation can be found in the author's published paper [20]). Quantum computation is probabilistic and the result of computation can be a probability distribution based on the inputs of the circuits. Hence, quantum circuit and quantum algorithms must be designed such that the correct result can be measured with high probability. Grover's search algorithm is used to evaluate the condition with very high measurement probability at an accelerated rate [19]. Assuming the system has N states labeled $S_1, S_2 \dots\dots S_N$ and there is a marked element S_m that satisfies a condition defined as C such that $C(S_m) = 1$ and for all other states $C(S_m) = 0$. Figure 6 provides a description of Grover's algorithm. Additional information about the implementation of Grover's search in this project can be found in the author's paper [20].

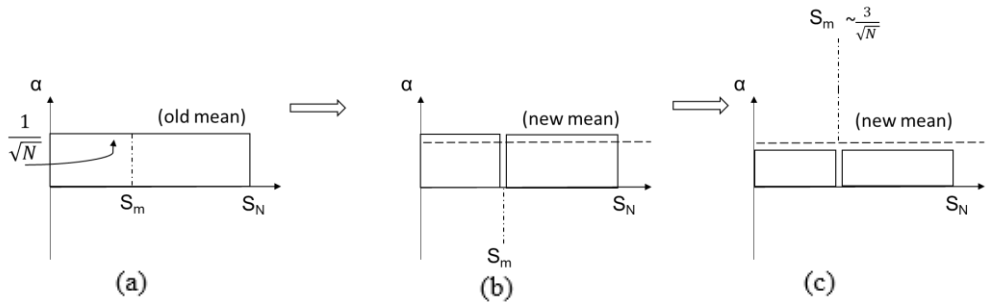


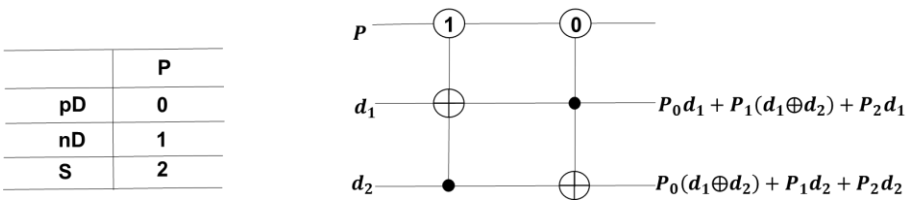
Figure 6: Grover's search algorithm illustration: (a) Quantum oracle identified the selected S_m; (b) invert the phase of the marked state C where $C(S_m) = 1$; (c) invert about the new mean for S_m to increase the amplitude α by $\sim\frac{2}{\sqrt{N}}$. The algorithm repeats the process a few times which is referred to as the Grover Diffusion operator [17]. After $O(\sqrt{N})$ operations (i.e., iterations $\left\lfloor\frac{\pi}{4}\sqrt{N}\right\rfloor$ where $\lfloor\ \rfloor$ is the floor function), the probability of measuring marked state approaches 100%. This is quadratic acceleration compared against classical search algorithm which needs *N* iterations to search unsorted data sets.

Finally, measurement can be done on the target register. From all superposed states, the marked have a significantly higher probability of measurement than the others and will most likely be selected. A random number generator is used in MATLAB simulation program to simulate the quantum measurement.

N states in quantum machine learning include training data and test data set for n-variables is: $N = 3^n \cdot 2^{(2^n - j)}$ where n denotes the number of input variables; $j \in (0,1,2, \ldots\ldots\ 2^n)$ is the number of the training ("known") data sets and $2^n - j$ is the number of values in the test data set (i.e., "target" has "don't know" value). Note that the training data set and test data set ratio is: $(j: 2^n - j)$.

2.3.2 The Quantum Oracle Circuit Design and Simulation

In this research, I designed a quantum circuit where the primary part is the quantum oracle necessary for Grover's Search. I created a novel polarity selection circuit to select KRO expansion, shown in Figure 7.a. The symbols in circles are quantum Toffoli Gates representing activating values of the quantum ternary qutrit *P*. The KRO butterfly in Figure 2 for three ways of expansions was implemented using two qubits (d_1 and d_2) as data inputs and a qutrit (P) as control inputs, where P_i value $i \in (0,1,2)$.



(a) (b)

Figure 7: Toffoli gate circuit synthesis for KRO expansion: when P =1, the first gate is activated ($d_{1\ \text{output}} = d_1 \oplus d_2$, $d_{2\ \text{output}} = d_2$); When P is 0, the second gate is activated ($d_{1\ \text{output}} = d_1$, $d_{2\ \text{output}} = d_1 \oplus d_2$); when P=2, none of gates is activated ($d_{1\ \text{output}} = d_1$, $d_{2\ \text{output}} = d_2$).

The matrix multiplication representation for Positive Davio, Negative Davio and Shannon Expansions as the base building block for a quantum gate in KRO processor:

$$M_{PD1} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{vmatrix} \qquad M_{ND1} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{vmatrix} \qquad M_{S1} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{10}$$

Assuming there is a 2^1 bit Boolean function, the inputs to these matrix multiplication in Equation (10) will be d_0d_1 = [00, 01, 10, 11]^T. For Positive Davio M_{PD1}: spectrum output e_0e_1 = [00, 01, 11, 10] ^{T} based on Equation (10) ($e_0=d_0$; $e_1 = d_0 \oplus d_1$); For Negative Davio M_{ND1}: the spectrum output e_0e_1 = [00, 11, 10, 01]^T based on Equation (10) ($e_0 = d_0 \oplus d_1$; $e_1=d_1$). For Shannon Expansion M_{S1}, the spectrum output e_0e_1 = [00, 01, 10, 11]^T for ($e_0 = d_0$; $e_1=d_1$) which is a pass through. This is how the Positive Davio, Negative Davio and Shannon Expansions can be converted into the matrix format in MATLAB. In addition, this also demonstrates how a quantum "superposition" processes all possible input values in parallel for accelerations.

In the KRO processor of the oracle from Figure 9 works by first assessing the polarity P_c with P_c= 1 (Negative Davio), 0 (Positive Davio), 2 (Shannon) in sequence; then P_b= 1 (Negative Davio), 0 (Positive Davio), 2 (Shannon) sequentially and finally P_a= 1 (Negative Davio), 0 (Positive Davio), 2 (Shannon) one after another from left hand side to right hand side. Given there are 3 (n =3) control inputs (P_a, P_b, P_c), the KRO oracle accepts the vector of input values $P_aP_bP_cd_0d_1d_2d_3d_4d_5d_6d_7$. Minterms d_i are constants 0 and 1, and "don't know" created by a $|0\rangle$ state followed by a Hadamard gate to create the "cat state" uniform superposition.

Figure 8 shows a novel 3-variable reconfigurable KRO Processor which was derived from the butterfly flow diagram and the KRO circuit synthesis realization from Figure 7 to find the optimum polarity

number and its corresponding maximal number of zero-valued KRO coefficients in a single, switchable butterfly structure.

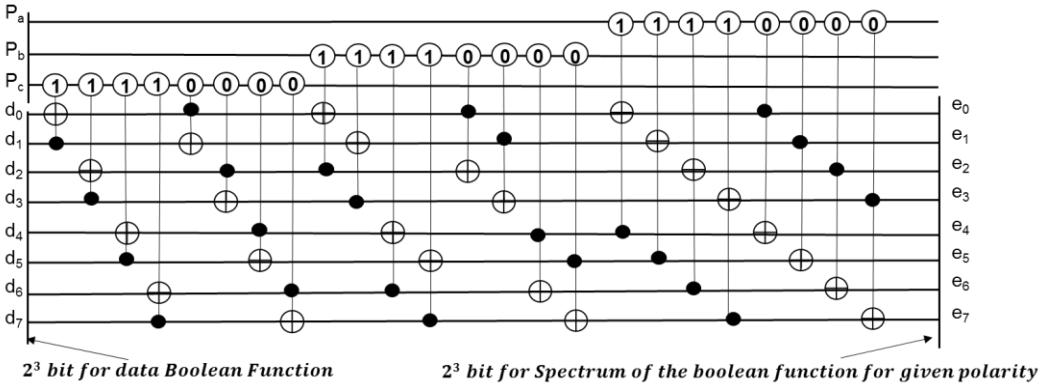


Figure 8: 3-variable ternary quantum KRO processors: Pa, Pb and Pc are 3 polarity of KRO expansions for 3-variables accordingly. $d_0d_1d_2d_3d_4d_5d_6d_7$ are 8 (2^3) minterms for the Boolean function f. Any of three expansions, pD, nD and S is realized in the quantum circuit implementation. The entire 8 columns for each Polarity (P_a, P_b and P_c) represents the butterfly KRO expansion for an individual variable x_1, x_2, x_3.

The proposed architecture of the quantum oracle is shown in Figure 9, composed of the 1) KRO processor, 2) Cost Function to calculate the nonzero coefficients 3) Comparator to compare against the threshold, and 4) the corresponding inverse blocks: Inverse KRO processor, Inverse Cost Function and Comparator. Please note inverse blocks restore quantum inputs so that inputs remain to be the original values for next Grover's search iteration inputs. These quantum gates introduced are reversible.

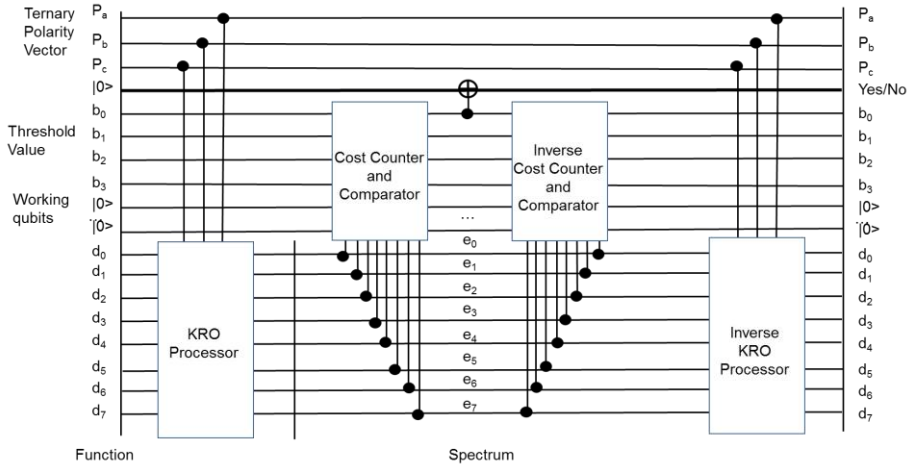


Figure 9: Quantum oracle architecture to find the minimum KRO forms based on Grover's Algorithm: Cost Function and Comparator are implemented to calculate the sum of non-zero coefficient for KRO

Figure 10 shows the flow chart for the quantum machine learning software. It entails: 1) The robot camera captures the image and feeds into MATLAB image process software 2) converts these image process results into input variable binary values 3) quantum oracle and Grover Diffusion operators are repeatedly applied to get the minimum KRO form for learning and then communicate to the robot for complex decision making (e.g., robot navigation directions, front arm actions, and body actions, etc.)

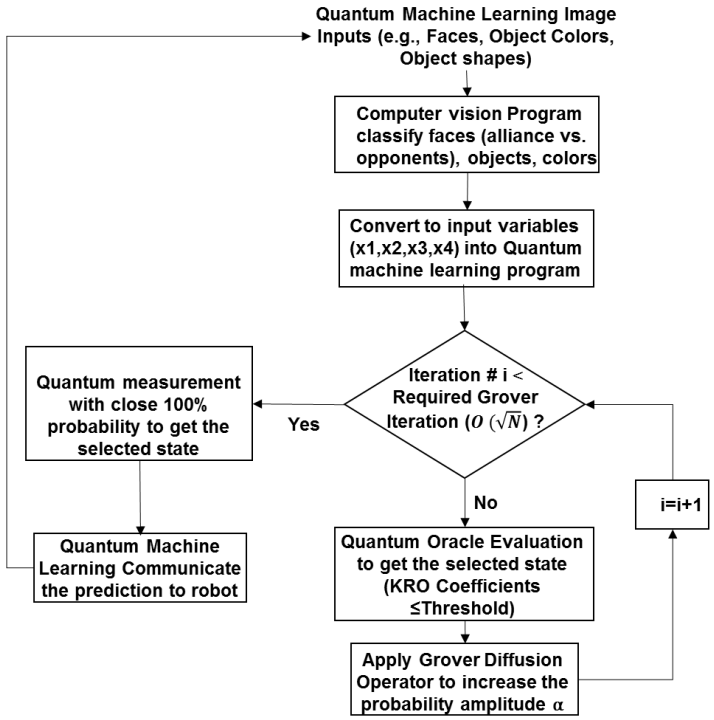


Figure 10: Quantum machine learning flow chart: Inputs from the robot camera go through image processing program before input into the machine learning program. After multiple iterations of Grover's search, quantum measurement will output the desired state *Sm* with ~100% probability. Please note P_a, P_b and P_c are ternary and $d_0 d_1 \ldots d_7$ are binary; hence the quantum oracle design is a ternary/binary hybrid quantum oracle ([21], [22] and [23]).

The unique innovations from my research include 1) Developed a novel Quantum Machine Learning algorithm for Boolean Function minimizations and demonstrated a computational decrease over classical machine learning methods 2) Increased Boolean function search space over previous FPRM forms with development of KRO expansions for optimized machine learning function 3) Created a novel hybrid quantum oracle for a unique application of Ternary and Binary hybrid Grover's Search 4) Invented a polarity selection circuit for KRO forms and which can be expanded for other RM forms 5) implemented the MATLAB simulation for this quantum machine learning algorithm to demonstrate its correctness.

2.4. Results

This algorithm was simulated using MATLAB. After applying quantum oracle and Diffusion operator multiple iterations ($i.e., \left\lfloor \frac{\pi}{4}\sqrt{N} \right\rfloor$ times), the output is displayed. 65,792 simulation runs for all possible 3 (2^8) and 4 (2^{16}) variable input minterm values were completed.

Table 1 shows samples of 4-variable KRO and FPRM Comparison for Machine Learning Scenarios with "Don't know" Inputs. It indicates that the non-zero coefficient sum for minimum KRO form versus minimum FPRM was reduced by 25-33%. This confirmed KRO forms can produce better or the same machine learning prediction functions with reduced non-zero coefficients.

Table 1: 4-variable KRO/FPRM Comparison for Machine Learning with "Don't knows" (-)

	Inputs Minterms	Polarity	Minimum KRO	1's	Polarity	Minimum FPRM	1's	Cost Reduction
1	10001001 1100110-	Pc=0 Pb=2 Pa=0 Pz=1	01000001 10000000	3	Pc=0 Pb=1 Pa=0 Pz=1	01000101 10000000	4	25%
2	10001001 1-001100	Pc=2 Pb=1 Pa=0 Pz=1	00000001 10000100	3	Pc=0 Pb=1 Pa=0 Pz=1	00000001 11000100	4	25%
3	11-01100 110011-1	Pc=0 Pb=2 Pa=0 Pz=0	10000000 00000001	2	Pc=0 Pb=0 Pa=0 Pz=0	10100000 00000001	3	33%
4	1000100- 110011-1	Pc=2 Pb=2 Pa=0 Pz=0	10000000 01000001	3	Pc=0 Pb=1 Pa=0 Pz=1	01000000 10000101	4	25%
5	01011101 - 10-11-1	Pc=2 Pb=1 Pa=0 Pz=0	00011000 00000000	2	Pc=0 Pb=1 Pa=0 Pz=0	00011100 00000000	3	33%
6	010-1111 - 1001-11	Pc=0 Pb=1 Pa=2 Pz=0	01000010 00000000	2	Pc=0 Pb=1 Pa=0 Pz=1	00000100 01000010	3	33%
7	0011-1-1 - 1-01110	Pc=0 Pb=0 Pa=2 Pz=0	00101000 01000001	4	Pc=0 Pb=1 Pa=0 Pz=0	10100100 00011000	5	20%
8	00-1-0-1 01001-10	Pc=2 Pb=0 Pa=0 Pz=0	00010000 01001000	3	Pc=0 Pb=0 Pa=0 Pz=0	00010000 01001001	4	25%

Figure 11 illustrates the Grover search acceleration for different ratio of training and testing "don't know" values. For example, the search can be accelerated by up to ~256 times for 5 variables when comparing quantum Grover search against equivalent classical search algorithms. The higher number of input variables will yield higher acceleration benefits.

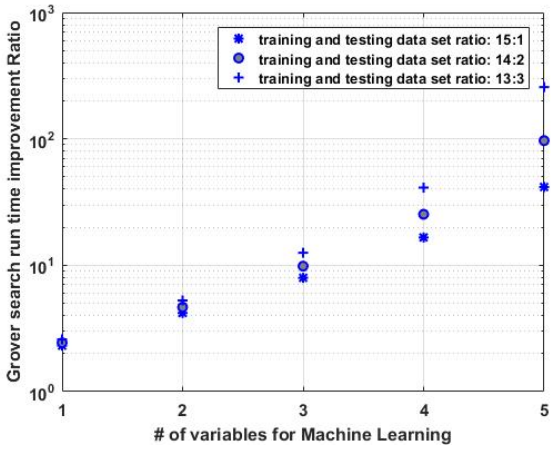


Figure 11: Grover acceleration analysis: Grover search will have more acceleration with the increase number of input variables. More "don't know" values will also have more Grover search accelerations.

As a case study, the intelligent robot decision making and recommendation for robot actions was analyzed to demonstrate that KRO minimum form has fewer non-zero spectral coefficients compared against FPRM forms. In Figure 12.a, it shows for the same incomplete Boolean function with multiple "don't know", the learned function with KRO is $\overline{x_3}\,\overline{x_4}\oplus x_4x_2$ while FPRM is $\overline{x_3}\,\overline{x_1}\oplus x_4x_2\oplus\overline{x_3}\,x_2$, which has more product terms and thus does not as closely adhere to Occam's Razor principle (note that product terms correspond to rectangles based on color in Figure 12.a). Figure 12.b shows the input HSV graph for the blue and red objects that robot needs to handle. The computer vision program that I developed can handle different combinations of object obstacles. Figure 12.c shows the Histogram of Oriented Gradients (HOG) feature example for a sample input face from the humanoid HR-OS1 robot camera input. This HOG feature will need to match some face classifiers from the human face database for the research purposes.

From the Karnaugh Map example in Figure 12.a, it is observed that when the input $x_4x_3x_2x_1$=1011, the KRO prediction is "1" and FPRM is "0". This shows that the KRO form adheres more closely to its surrounding cells in the Karnaugh map (i.e., change from $x_4x_3x_2x_1$=1011 to $x_4x_3x_2x_1$=1111, 1010 or 1001), resulting in more generalizable predictions. This is aligned with the fact that KRO covers the "don't know" with fewer rectangles. As a result, the robot prediction function is simpler and also saving the computational cost when new input data are gathered from the robot. The benefit is significant where multiple Boolean functions are needed to govern complex application system behavior.

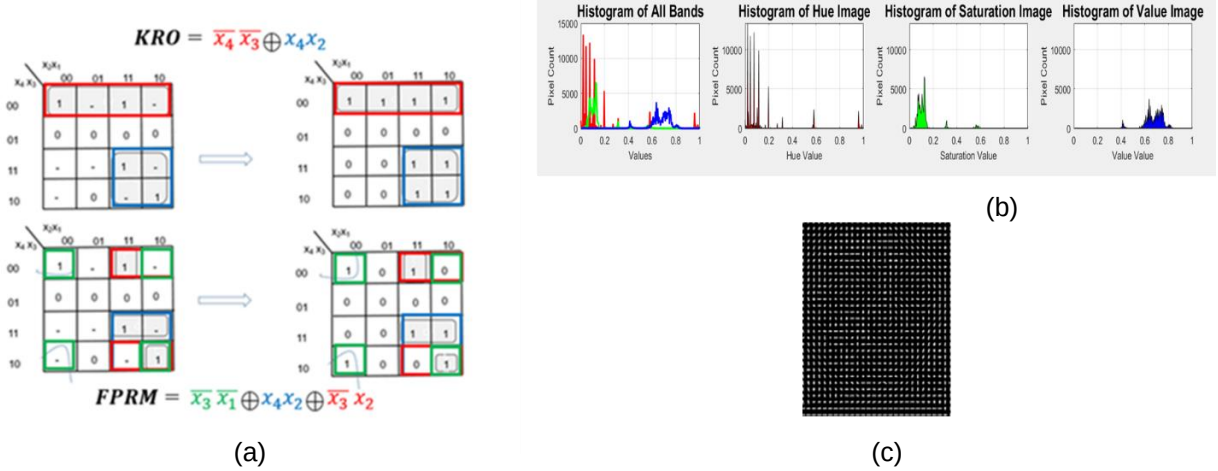


Figure 12: Illustration of KRO form vs. FPRM form quantum machine learning example (a) 4 variable minimum KRO vs. minimum FPRM forms (b) Input image data (HSV) color representation for red and blue objects (c) Face HOG feature captured from digital camera mounted on humanoid HR-OS1 robot.

3. Conclusions and Future Work

A novel Grover-based quantum computing program was designed and simulated for machine learning based on KRO form minimization for the first time. The correctness of the quantum algorithm was validated via 65,792 MATLAB simulation runs. Moreover, 600 simulation runs were also completed with "don't know" for machine learning use cases to cover different input parameters.

In this research, scenarios with "don't knows" were simulated in MATLAB. In the machine learning examples with "don't knows" from this research, it shows up to ~33% improvement. In addition, this research validated Grover search quadratic acceleration and indicates the acceleration benefits will be higher with more input variables. Finally, the case study was analyzed for intelligent robot quantum machine learning scenarios and proved KRO yields better prediction functions.

My future research plan in quantum machine learning algorithm and implementation area includes: (1) accelerate the algorithm using computer clusters to simulate or multiple FPGA (Field Program Gate Array) boards to emulate quantum operations (Negovetic et al. [24]) to validate for more applications; (2) extend this system to full multiple-valued beyond ternary logic (e.g., qudits) to further decrease the quantum cost of the circuit, (3) Research other quantum search algorithm based on families of spectral butterflies, e.g. GRM (Generalized Reed-Muller Forms) or ESOP (Exclusive-OR Sum of Products), to further accelerate search and decrease the complexity of machine learning prediction solutions.

References:

1. Samuel, Arthur; *Some Studies in Machine Learning Using the Game of Checkers.* IBM Journal of Research and Development, Volume: 44, Issue: 1.2. DOI:10.1147/rd.441.0206
2. Le, Quoc V.; Monga, Rajat; Devin, Matthieu; Chen, Kai; Corrado, Greg S.; Dean, Jeff; and Ng, Andrew Y.; *Building High-level Features Using Large Scale Unsupervised Learning*, International Conference on Machine Learning, p.103, 2012
3. Abbeel, Pieter; *Machine Learning for Robotics*, Keynote speech in European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), Bristol 2012
4. Di, Y. and Wei, H.; *Elementary Gates of Ternary Quantum Logic Circuit*, Quantum Physics, April, 2012.
5. Rebentrost, M. Mohseni and Lloyd, S.; *Quantum Support Vector Machine for Big Data Classification*, Phys. Rev. Lett., 113 (130503), 2014.
6. Yoo, S.; Bang, J.; Lee, C. and J. Lee; *Quantum speedup in machine learning: finding an N-Bit Boolean function for a classification*, New Journal of Physics, vol. 16, pp1-15, October 2014
7. Cai, X.; Wu, D.; Su, Z.; Chen, M.; Wang, X.; Li, L.; Liu, N.; Lu, C. and Pan, J.; *Entanglement-Based Machine Learning on a Quantum Computer*, In Quantum Physics, Vol. 114, Iss. 11 — 20 March 2015.
8. Lloyd, Seth; Mohseni, Masoud; and Rebentrost, Patrick; *Quantum algorithms for supervised and unsupervised machine learning*, arXiv preprint arXiv:1307.0411, 2013.
9. Li, L.; Thornton, M.; and Perkwski, M.; *A Quantum CAD Accelerator Based on Grover's Algorithm for Finding the Minimum Fixed Polarity Reed-Muller Form*, 36th International Symposium on Multiple-Valued Logic, 2006.
10. Sarabi, A.; Perkowski, M.A.; *Fast Exact and Quasi-Minimal Minimization of Highly Testable Fixed-Polarity AND/XOR Canonical Network*, Proc. DAC, 1992, pp. 30-35.
11. Drechsler, R.; Becker, B.; Göckel, N.; *A Genetic Algorithm For Minimization of Fixed Polarity Reed-Muller Expressions*, In IEEE Proceedings Computers and Digital Techniques, Vol. 143, pp. 364-368, 1996.
12. HR-OS1 Humanoid Robot Overview: *http://learn.trossenrobotics.com/38-interbotix-robots/164-hr-os1-humanoid-overview.html*
13. Davio, M.; Deschamps, J.P.; Thayse, A.; *Discrete and Switching Functions*, McGraw Hill, 1978.
14. *Soklakov, A.N.; Occam's Razor as a formal basis for a physical theory*, Foundations of Physics Letters. Springer. 2002.
15. Kalay, U.; Hall, D. V.; and Perkowski, M.; *Easily Testable Multiple-valued Galois Field Sum-of-Products circuits*, Journal on Multiple Valued Logic, 2000, Vol. 5, pp. 507-528.
16. HOG Document: *http://www.mathworks.com/help/vision/ref/extracthogfeatures.html* (Mathworks)
17. Nguyen, T.; *Shaat*, G.; Schaeffer, B.; Truong, J.; and Upperman, N.; *Intelligent robotics*, Portland State University research report. Winter 2011.
18. Nielsen, M.A. and Chuang, I. L., *Quantum Computation and Quantum Information*, Cambridge University Press, 2000.

19. Grover, L.; *A Fast Quantum Mechanical Algorithm for Database Search*, Proceedings of the 28th Annual ACM Symposium on Theory of Computing, 1996, pp. 212-219.
20. *The Competition Entrant* and *The Competition Entrant's mentor*; *Quantum Machine Learning Based on Minimizing Kronecker Reed-Muller Forms and Grover Search Algorithm with Hybrid Oracles*, Proceeding of EUROMICRO Conference on Digital System Design, August, 2016, pp.413-422
21. Yang, G.; Hung, W.N.N.; Song, X.; Perkowski, M.A.; *Exact Synthesis of 3-qubit Quantum Circuits from Non-binary Quantum Gates using Multiple-Valued Logic and Group Theory*, Research Report, Portland State University, Portland, Oregon, 2004.
22. Wang, Y.; Perkowski, M.; *Improved Complexity of Quantum Oracles for Ternary Grover Algorithm for Graph Coloring*, Proceeding of 41st IEEE International Symposium on Multiple-Valued Logic, pp.294-301, 2011.
23. Mandal, S. B.; Chakrabarti, A.; and Sur-Kolay, S.; *Synthesis of Ternary Grover's Algorithm*, IEEE 44th International Symposium on Multiple-Valued Logic, 2014.
24. Negovetic, G.; Perkowski, M.; Lukac, M.; Buller, A.; *Evolving Quantum Circuit and an FPGA-based Quantum Computing Emulator*, Research Report, Portland State University, Portland, Oregon, 2002.