International Journal of Communication Networks and Information Security
2024, 16(4)
ISSN: 2073-607X,2076-0930
https://ijcnis.org/

Research Article

The Revolution of Quantum Computing: Analyzing Its Effects on Cryptographic Security and Algorithmic Efficiency

Thilagavathy R[1], Gayathri M[2,*]
[1,2] Department of Computing Technologies, College of Engineering and Technology, SRM Institute of Science and Technology, Kattankulathur, Tamilnadu, India.
*Correspondence: gayathrm2@srmist.edu.in

ARTICLE INFO

Received: 10 Aug 2024
Accepted: 16 Sep 2024

ABSTRACT

This study examines how quantum computing influences cryptographic security and algorithmic performance. Quantum computers, using qubits, superposition, and entanglement, present dangerous levels of risks to traditional cryptography and show promising potential for calculated speed. For predicting secure and efficient quantum computing, the current study adopts some machine learning models such as Decision Trees, Random Forests, and K-nearest neighbours. The Decision Tree model is the most accurate model with 100% precision, and there is a need to incorporate studies and research on quantum-safe algorithms and post-quantum cryptography for potential risks brought by quantum in the future.

Keywords: Quantum computing, cryptographic security, algorithmic efficiency, machine learning models, post-quantum cryptography, Decision Tree, quantum-safe algorithms, quantum noise, quantum algorithms.

I. INTRODUCTION

1. Background

Quantum computing is introducing a new notion of computing and of processing capabilities into the field of computational science by questioning the security and cryptography and increasing the enhancement of algorithms.While traditional computers employ 'bits,' quantum computers have 'qubits' and utilize the concepts of superposition and entanglement for solving problems. This is more of a technological advancement that comes with its benefits and drawbacks, especially in areas such as cryptography where existing methods of encryption might be at risk [1]. Quantum algorithms, such as Shor's and Grover's, do provide, in concept, a possibility for enhanced power of computation that might transform industries reliant on massive computation and secure data transfers. It is within this framework that the said effects are to be described in greater detail in this research.

2. Aim and Objectives

Aim

The main purpose of this study is to impacts of quantum computing on cryptographic security and algorithmic efficiency using key parameters and trend analysis, supported by machine learning models.

Objectives

- To assess the relationship between quantum computing parameters, such as execution time and data size.
- To evaluate the impact of quantum noise levels on algorithmic performance.
- To compare the effectiveness of various machine learning models in predicting secure and efficient quantum computing outcomes.

- To Identification of potential vulnerabilities in existing cryptographic systems due to the development of quantum systems.
- To provide insights into optimizing algorithms in a quantum computing environment.

II. LITERATURE REVIEW

2.1: Quantum Computing Basics and Development

Quantum computing is a revolutionary technology that continues the advances in classical computing, but instead of a classical approach, it uses quantum mechanics as the base technology [2]. Therefore, at the heart of quantum computing are the so-called qubits, which can be considered as quantum versions of the classical bits. While there exist only two options – "0" and "1" in bits, qubits utilize quantum processes like superposition and entanglement. Superposition means that a qubit, at the same time, is in states 0 and 1; entanglement means that qubits can be correlated in a manner that is impossible with classical bits. Quantum computing is a subfield that emerged in the early 1980 s with the theoretical works of Richard Feynman and David Deutsch who argued that the laws of quantum mechanics could be used to perform computations.

Time Complexity of Classical vs. Quantum Algorithms

$$T_{quantum}(n) \leq O(log^k n) \cdot Tclassical(n)$$

The real-world implementation of QC has been through different stages depending on certain achievements. Some of the early experiments include; the realization of basic quantum gates and simple quantum algorithms on quantum processors. Some of these developments have been pioneered by quantum leaders such as IBM, Google, Microsoft and other players such as D-Wave, and Rigetti Computing among others. Google claims to have achieved "quantum supremacy' in 2019 where a quantum computer solved a certain problem in a smaller amount of time than any other classically existing supercomputer [3]. This point highlighted the advancement and potential of quantum systems but large-scale quantum computation is still a dream. Quantum Computing Hardware has also evolved significantly with different architectural techniques like superconducting bits, trapped ions, and topological bits with their advantages and disadvantages. With further studies, the major trends are shifting to expanding the physical size of quantum systems, boosting qubit coherence times and enhancing methods of error correction to enable precisely using quantum computing for solving diverse challenges and enriching different sectors at large.

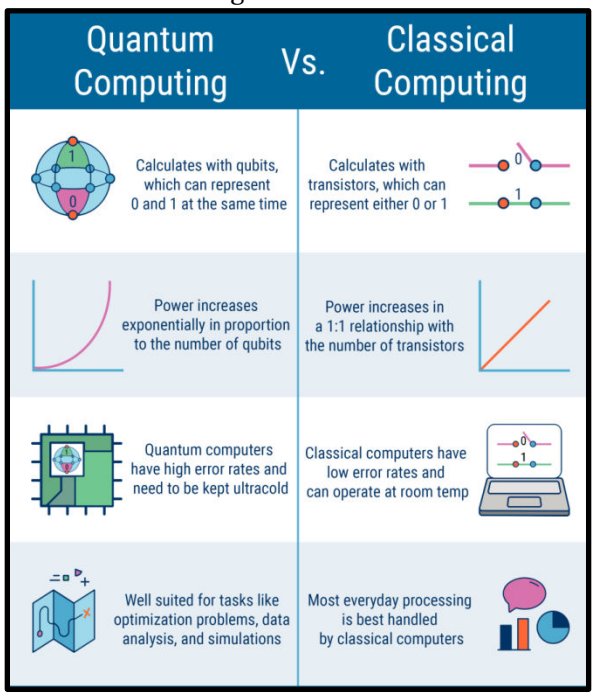


Fig 1: Quantum Computing vs Classical Computing

2.2: **Impact of Quantum Computing on Cryptographic Security**

The main threat that quantum computing has on classical cryptography is its ability to break most of the commonly used cypher algorithms. That is why symmetric and asymmetric cryptographic systems, for example, RSA, ECC, and AES, are based on mathematical theorems and problems which are virtually

insolvable for classical computers and are considered to be reasonable. However, quantum computers utilise different quantum algorithms that are able to solve these problems much faster in terms of exponential way. One of the most well-known algorithms that have significant effects on cryptographic security is Shor's algorithm which had been developed by Peter Shor in 1994 [4]. Shor's algorithm effectively factors a large integer and computes discrete logarithms thus threatening the security of RSA and ECC. For example, by using an extremely large semiprime number it takes a classical computer a huge amount of time to factor these numbers and thus break RSA encryption, while a polynomial time can be provided by a sufficiently powerful quantum computer thus making RSA encryption easily crackable.

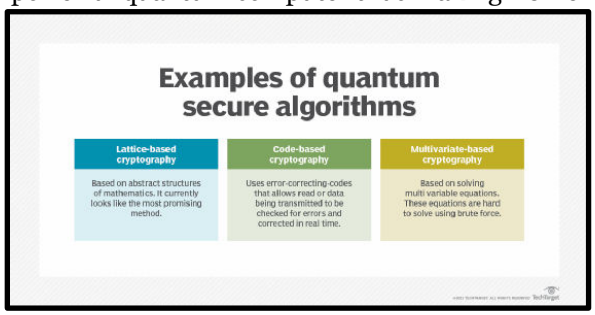


Fig 2: Secure Algorithms

Grover's algorithm is also another quantum algorithm that poses a threat to symmetric-key cryptography including the AES. Grover's algorithm offers a quadratic speed-up for unstructured search problems, which may even cut down the effective key length of the symmetric encryption methods in the ratio of 1/2. This means that AES-128 for instance would provide the same security as AES-64 against a quantum adversary for the use of longer keys for security [5]. The effects of these advancements have led to the development of what is referred to as post quantum cryptography which is a form of cryptographic system that is safe from both classical and quantum cryptography attacks. The research is still being undertaken to find algorithms that cannot be vulnerable to quantum attacks, including lattice, hash, and code-based systems. The migration procedure to post-quantum cryptography is necessary as quantum computing develops for protecting valuable data.

2.3: Algorithmic Efficiency and Quantum Computing

Quantum computing presents shift advancements in the speed of the algorithmic compared to the classical computers because of quantum mechanical theory. That potential is exemplified by one of the elementary quantum algorithms — the Quantum Fourier Transform (QFT) [6]. QFT is an important part of some algorithms such as Shor's where large integers can be factored in polynomial time. This can be considered a breakthrough as compared to classical methods that work in exponential time in order to solve similar problems, which greatly increases the effectiveness of problem-solving in number theory and cryptography. Another quantum algorithm that has attracted the attention of researchers is Grover's algorithm which gives quadratic speed up for searching in unsorted databases.

Shor's Algorithm Complexity

$$T_{quantum}(n) = O((logn)^2(loglogn)(logloglogn))$$

Grover's Algorithm Complexity

$$T_{quantum}(n) = O\sqrt{(N)}$$

Whereas classical algorithms take linear time in searching through n items Grover's algorithm can do so in approximately the order of √n. Amortization helps, though as Nagel and Stevens demonstrated, it is not as radical an improvement over a brute-force solution as Shor's exponential speedup; however, for certain classes of problem, such as optimization and data-mining, the gain is substantial [7]. Quantum computing also improves the algorithms' effectiveness in many other areas of application. For instance, quantum computers have the capability to perform what is known as quantum simulations of quantum systems which hold the promise of more accurate and efficient solution of physical and chemical problems as compared to classical simulations.

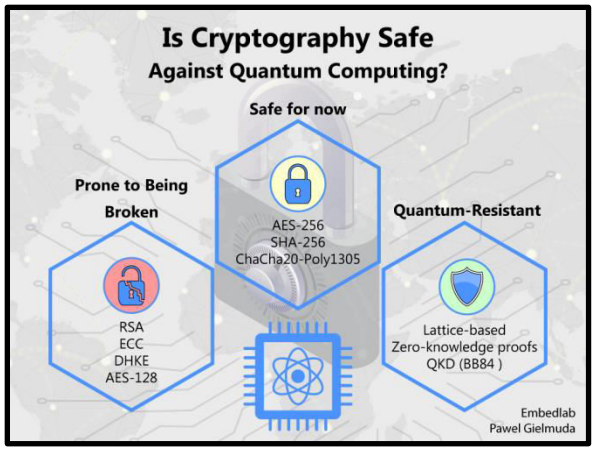


Fig 3: Analysis of the Safety of Quantum Computing

This capability poses incredible ramifications to such sectors as material science and drug discovery since quantum interaction must be modelled with precision. However, this efficiency, increases are yet to be realized practically due to the improvements in the quantum hardware and error-correction systems [8]. These efficiencies can only be today shown on small-scale quantum computers due to the limitations with current numbers of qubits and tolerable error rates.

Quantum Entropy

$$S(\rho) = -Tr(\rho log \rho)$$

2.4: Future Directions and Challenges

The development of quantum computing is one of the most fascinating topics, as well as one filled with great opportunities and imminent problems. Over the years as the field advances, some directions and challenges are becoming apparent [9].

1. Scaling Up Quantum Systems: Quantum computers' problem-solving ability at present is highly limited; one of the major issues is to increase the number of quantum bits. It is significant for the current quantum systems to be aware of the number of qubits involved and their error rates. To reach a quantum advantage it is necessary to fabricate quantum processors with thousands and millions or more qubits with high fidelity and coherence. Many types of implementations are being researched, such as superconductive qubits, trapped ions and topological qubits, which means that each has its own particular technical implications [10].

2. Error Correction and Stability: However, it is significant to note that quantum error correction is essential in order to achieve reliable quantum computation. In contrast to the classical systems, quantum states of information are very sensitive to the noise originating from decoherence as well as from operation inaccuracies. The precise error correction codes and the stabilization of these quantum states can be regarded as the key factors that determine large-scale quantum computers. A substantial amount of work is dedicated to increasing error rates and designing reliable quantum computing systems.

3. Quantum Algorithm Development: There are still many questions as to how quantum computing could be harnessed, and the search for further quantum algorithms is its active refining [11]. Even though algorithms like Shor's and Grover's do exist, they have to be practical founded algorithms relevant to a specific sector of industry and problem domain. Further development of quantum algorithms: It is demonstrated that QAOA has further reaching implications for optimization, machine learning as well as simulation applications of quantum computing than the traditional Shor's algorithm, Likewise, more quantum algorithms are required to harness the full potential behind quantum computing.

4. Post-Quantum Cryptography: Thus, in parallel with the emergence of practically usable quantum computers, the need to switch to post-quantum cryptographic methods becomes more pressing. Scientists have started trying to design techniques that will eventually replace today's cryptographic algorithms and are in the process of establishing post-quantum cryptography algorithms.

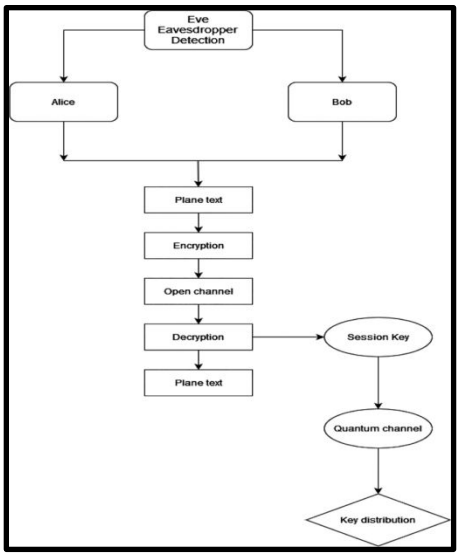


Fig 4: Flowchart of quantum cryptosystem

2.5: Literature Gap

Although significant research has been carried out to understand the theoretical foundations and application of quantum computing especially in the area of cryptography and algorithms, there are still large voids that have not been closed. Namely, there is the absence of systematic investigations concerning the key issues affecting the practical application of post-quantum cryptographic systems and their application in unified structures. Further, the studies are mostly centered on individual quantum algorithms and do not consider their versatility in varying sectors or integration with the upcoming quantum technologies.

III. METHODOLOGY

1. Data Collection and Preprocessing

Quantum Speedup

$$S(n) = \frac{T_{classical}(n)}{T_{quantum}(n)}$$

The data for this study is gathered from multiple research areas that are relevant to quantum computing, cryptography security and algorithm efficiency. Such values are the program's run time, the energy consumed, the probability of errors, levels of quantum noise, the type of the algorithm, the complexity of the algorithm, and the volume of data [12]. These parameters are valuable in assessing the security and efficacy of quantum computing platforms. It first has to undergo a cleaning process to ensure that the data is suitable for analysis. This includes how missing values are handled, outliers are handled and categorical data is dealt with to enable the modelling phase. One process is normalization and standardization, which is done to adjust the scale of the feature so that it can help the efficiency of the machine-learning algorithms. The dataset is then divided into training and testing data sets for the models so that the efficiency of the models must be tested while training them. These preprocessing steps assist in standardizing the data to enhance the subsequent analysis.

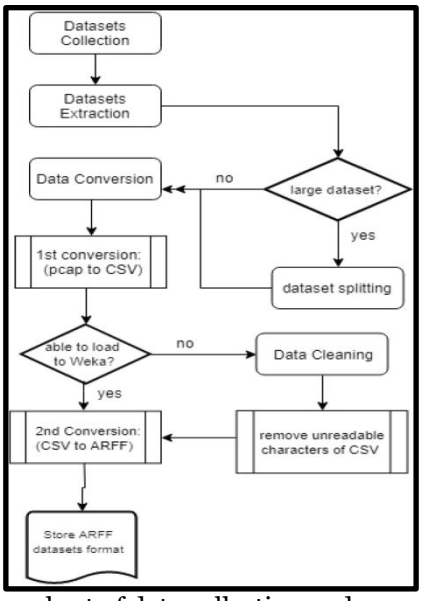


Fig 5: Flow chart of data collection and preprocessing

Quantum Error Rate

$$\epsilon = 1 - (1 - \epsilon_0)^N$$

2. Exploratory Data Analysis (EDA)

In this study, EDA is done to investigate the correlation between important quantum computing variables. A range of representations, including scatter plots, heatmaps, histograms, and violin plots, are used in the study to explore the distribution and interdependency of features like execution time energy consumption and quantum noise levels [13]. This analysis has shed more light on the interdependency of various factors that are of importance in cryptographical security and algorithmic optimization in quantum computing.

3. Machine Learning Models

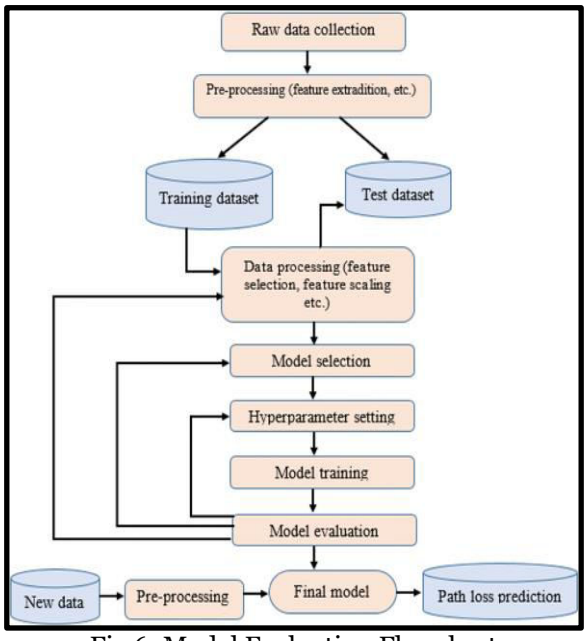


Fig 6: Model Evaluation Flowchart

Logistic Regression

This paper may utilize Logistic Regression as the first method of prediction because it is easy to implement and provides output with clear interpretations. It outputs the likelihood of the given quantum computing setup belonging to the secure/efficient class by fitting the input features through a logistic

function [14]. It is especially useful to determine the coefficients of the linear regression of the quantum computing parameters about security/efficiency classification.

1. START

2. LOAD the dataset 'quantum_computing_dataset.csv'

3. PREPROCESS the data:
a. HANDLE missing values and outliers if present.
b. CONVERT categorical variables (e.g., 'Algorithm Complexity', 'Algorithm Type') to numerical codes.
c. SPLIT the data into features (X) and target (y) where:
- X contains all columns except 'Secure/Efficient'
- y contains the 'Secure/Efficient' column (the target variable).
d. STANDARDIZE the features (X) using StandardScaler to normalize the data.

4. SPLIT the standardized data into training and testing sets:
a. SET the training set to 70% of the data.
b. SET the testing set to 30% of the data.

5. INITIALIZE the Logistic Regression model.

6. TRAIN the Logistic Regression model using the training set (X_train, y_train).

7. PREDICT the target values (y_pred) using the trained model on the testing set (X_test).

8. EVALUATE the model's performance:
a. CALCULATE the accuracy of the model by comparing y_test with y_pred.
b. GENERATE a classification report to obtain precision, recall, and F1-score.

9. OUTPUT the accuracy and classification report.

10. END

Decision Tree Classifier

The Decision Tree Classifier is used to fit the given dataset in a tree-based structure. This model works by recursively dividing the data set based on the features followed by a branch which leads to a classification judgement. Every vertex in the tree is a feature and the edges connecting the vertices are the possible values of the feature. The main benefits of the Decision Tree model include handling both numeric and categorical data and improving the decision-making process through the visual flow diagram [15].

1. Start with the entire dataset as the root node.

2. Check if all data points in the node belong to the same class:
a. If yes, label the node with that class and stop further splitting.
b. If no, proceed to step 3.

3. For each feature in the dataset, calculate the following:
a. Information Gain (IG) or Gini Impurity for splitting based on that feature.

4. Select the feature with the highest Information Gain or lowest Gini Impurity.

5. Split the dataset into subsets based on the selected feature's possible values:
a. Each subset becomes a child node.

6. Repeat steps 2 to 5 recursively for each child node:
a. Continue splitting until:
i. All data points in a node belong to the same class.
ii. No further information gain is possible.
iii. A pre-defined stopping criterion (like maximum depth) is met.

7. Assign the class label that corresponds to the majority class of the data points in each leaf node.

8. Once all nodes are processed, the Decision Tree is complete.

9. To classify a new data point:
a. Start at the root node.
b. Traverse the tree by following the path determined by the feature values of the data point.
c. Continue until reaching a leaf node.

d. Assign the class label of the leaf node to the data point.

10. End.

Random Forest Classifier

Random Forest is another machine learning technique that is derived from decision trees, where multiple decision trees are built to improve the classification accuracy of the model. It uses randomly selected subsets of the data to create many different decision trees and then provides the average of their predictions [16]. Also, Random Forest allows for determining feature importance, which is useful for identifying the most significant characteristics that affect the security and effectiveness of quantum computing arrangements.

1. Initialize the Random Forest Classifier
- Set the number of decision trees (n_estimators) to be created
- Define other parameters such as max_depth, min_samples_split, etc.

2. For each decision tree in the Random Forest:
a. Randomly select a subset of the training dataset (with replacement)
- This subset is called a bootstrap sample
- Ensure each tree gets a different bootstrap sample

b. For each decision tree:
i. Randomly select a subset of features to be used for splitting nodes
- This subset is called a feature subset
- Choose the best feature from this subset based on a splitting criterion (e.g., Gini impurity, information gain)

ii. Build the decision tree by recursively splitting the nodes:
- Start at the root node with all data points from the bootstrap sample
- At each node, choose the best feature and threshold to split the data
- Continue splitting until a stopping condition is met (e.g., max_depth, min_samples_split)

iii. Store the trained decision tree

3. After training all the decision trees:
a. For a new data point (test instance):
i. Pass the data point through each decision tree
ii. Each decision tree makes a prediction (class label)

b. Aggregate the predictions from all decision trees:
- Perform majority voting to determine the final class label
- The class label with the most votes is chosen as the final prediction

4. Output the final prediction for the data point

5. Evaluate the overall model:
- Compute accuracy, precision, recall, and F1-score on the test dataset
- Analyze feature importance to understand which features contribute the most to the model's decisions

End of Pseudocode

Support Vector Machine (SVM)

SVM is used due to its ability to find the optimal hyperplane of the distinction between secure/ efficient and insecure/inefficient environments. SVM is most suitable when data can be transformed into a high dimension space which is advantageous in the research wanting to look into a variety of features incorporated in this study [17]. The model works on the principle of trying to maximize the distance between the two classes with the intent of categorizing the setups with as much certainty as possible. This evaluation metric is good for SVM due to its absence of overfitting and versatility in handling non-linearity in the decision surface.

1. Import necessary libraries
- Import SVM module from sklearn
- Import train_test_split, StandardScaler, and accuracy_score from sklearn

2. Load and preprocess the data
- Load the dataset
- Split the dataset into features (X) and target (y)
- Split the data into training and testing sets using train_test_split
- Standardize the features using StandardScaler

3. Initialize the SVM model
- Create an instance of the SVM classifier (e.g., SVC())

4. Train the SVM model
- Fit the SVM model on the training data (X_train, y_train)

5. Make predictions
- Use the trained model to predict the target values for the test set (X_test)

6. Evaluate the model
- Calculate the accuracy of the model using accuracy_score
- Optionally, generate a classification report and confusion matrix

7. Output the results
- Print the accuracy score and other evaluation metrics

K-Nearest Neighbors (KNN)

KNN is an unsupervised machine learning algorithm that categorises a quantum computing setup by training it on the most occurring class of its closest neighbours in the feature domain. KNN is easy to implement and works well especially when the classes do not have a linear separation [18]. This makes the model rather interpretable since the distances between the data points are directly taken into consideration for the classification.

1. Input: Dataset D with features X and labels y, new data point x_new, number of neighbors k.

2. Preprocess:
a. Standardize or normalize the features in D to ensure consistent distance calculations.

3. For each data point xi in the dataset D:
a. Calculate the distance between xi and x_new using a distance metric (e.g., Euclidean distance).
b. Store the distance and the label of xi.

4. Sort the distances in ascending order.

5. Select the top k nearest neighbors to x_new.

6. Aggregate the labels of the k nearest neighbors (e.g., majority voting for classification).

7. Assign the most common label among the k neighbors to x_new.

8. Output: Predicted label for x_new.

4. Model Evaluation and Validation

In this analysis, the key performance metrics of the machine learning models are chosen about accuracy, precision, recall, and F1-score. These metrics will give full profiling of the performance of each model in classifying the quantum computing setups into secure/efficient or insecure/inefficient classes. In addition, there is a comparison of performance for the models based on score accuracy and an analysis of feature importance for the best-performing model. This goes a long way in ensuring that the output classifications are reliable and robust [19].

5. Quantum Communication Methodology

The general method of operation of quantum communication is based on principles of quantum mechanics. One of the primary means is in Quantum Key Distribution or QKD which is the usage of quantum states to exchange cryptographic keys owing to physical realities such as no-cloning theorem and quantum entanglement. Eavesdropping is possible in the BB84 protocol that uses photon polarization to transmit keys while any such attempt must be immediately detected. This methodology ensures that each interception changes the state of the quantum state making the communicating parties aware of security threats and therefore, has a central role in the progress of secure communication in the age of quantum technology.

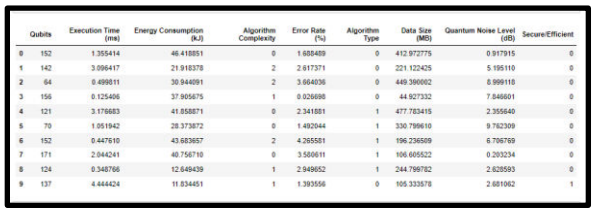
6. Quantum Cryptography Protocols

Quantum cryptography protocols are essentially meant for the protection of information that transmits through the management of quantum mechanical features. Such protocols as BB84 and E91 employ quantum characteristics, including entanglement and the no-cloning theorem to generate keys that

cannot be deciphered with classical or quantum hacks. Another advanced form of QKD is the Continuous Variable Quantum Key Distribution (CV-QKD) which employs continuous variables rather than discrete ones, especially in complex conditions. These protocols make up the framework of quantum cryptography so that anyone who tries to eavesdrop or meddle with the transmission must be noticed.

IV. RESULT AND DISCUSSION

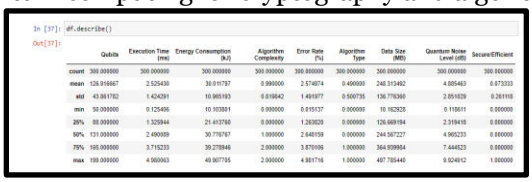
1. Result



	Qubits	Execution Time (ms)	Energy Consumption (kJ)	Algorithm Complexity	Error Rate (%)	Algorithm Type	Data Size (MB)	Quantum Noise Level (dB)	Secure/Efficient
0	152	1.355414	46.418851	0	1.688489	0	412.972775	0.917915	0
1	142	3.096417	21.918378	2	2.617371	0	221.122425	5.195110	0
2	64	0.499811	38.944091	2	3.664036	0	449.390002	8.999118	0
3	156	0.125406	37.905675	1	0.026698	0	44.927332	7.846601	0
4	121	3.176683	41.858871	0	2.341881	1	477.783415	2.355640	0
5	70	1.051942	28.373872	0	1.492044	1	330.799610	9.762309	0
6	152	0.447610	43.683657	2	4.265581	1	196.236509	6.706769	0
7	171	2.044241	40.756710	0	3.580611	1	106.605522	0.203234	0
8	124	0.348766	12.649439	1	2.949652	1	244.799782	2.628593	0
9	137	4.444424	11.834451	1	1.393556	0	105.333578	2.681062	1

Fig 7: First few rows of the dataset

This figure illustrates the Sample of the dataset used in the study and the few features that are defined or being analyzed in the study. The variables in the dataset include Qubits, Execution Time (ms), Energy Consumption (kJ), Algorithm Complexity, Error Rate (%), Algorithm Type, Data Size (MB), Quantum Noise Level (dB), and Secure/Efficient which is the target variable. The data is necessary to analyze the potential consequences of quantum computing for cryptography and algorithm performance.



```
In [37]: df.describe()
Out[37]:
```

	Qubits	Execution Time (ms)	Energy Consumption (kJ)	Algorithm Complexity	Error Rate (%)	Algorithm Type	Data Size (MB)	Quantum Noise Level (dB)	Secure/Efficient
count	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000
mean	129.916667	2.525430	30.011797	0.990000	2.574874	0.490000	248.313492	4.885463	0.073333
std	43.861782	1.424291	10.965193	0.818642	1.491977	0.500735	136.776360	2.851029	0.261118
min	50.000000	0.125406	10.103881	0.000000	0.015137	0.000000	10.162926	0.118611	0.000000
25%	88.000000	1.329944	21.413768	0.000000	1.263828	0.000000	126.881194	2.319418	0.000000
50%	131.000000	2.490089	30.778787	1.000000	2.648159	0.000000	244.567227	4.965233	0.000000
75%	165.000000	3.715233	39.278846	2.000000	3.870106	1.000000	364.939864	7.444523	0.000000
max	199.000000	4.980963	49.907795	2.000000	4.981716	1.000000	497.785440	9.924912	1.000000

Fig 8: Description of the dataset

This figure shows descriptive statistics of the dataset. They summarize central tendencies, dispersions, or shapes regarding the distribution of the dataset. It includes measures like count, mean, standard deviation, minimum, maximum, and percentiles for each feature [20]. The mean number of Qubits is around 129.92, while the mean Quantum Noise Level is around 4.88 dB. Such statistics help in understanding the patterning and overall variability in the data.

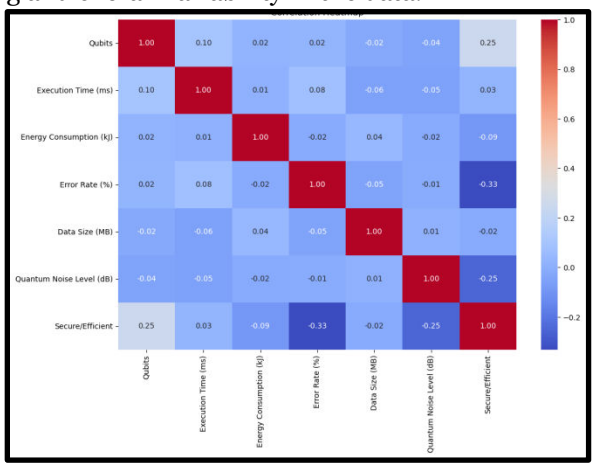


Fig 9: Correlation Heatmap

The correlation heat map gives a relation between various numerical features of the dataset. Dark red in the heat map signifies strong correlations, while blue reflects weak ones. Qubits and Secure/Efficient both show a moderate positive correlation of 0.25, however, it is very negatively correlated between Error Rate (%) and Secure/Efficient, at -0.33. Hence, this heat map is useful for detecting features that could have more weight toward the target variable.

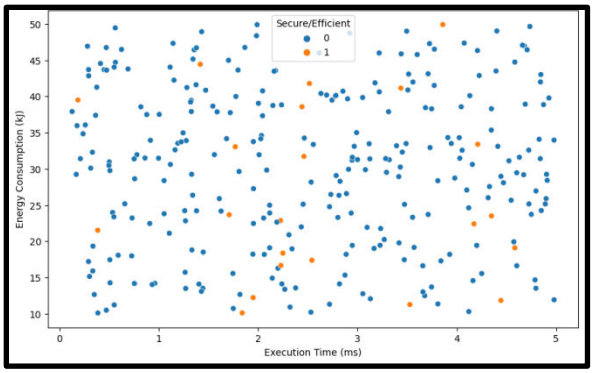


Fig 10: Scatter Plot of Execution Time vs Energy Consumption

This scatter plot also shows the corresponding Execution Time in milliseconds as well as Energy Consumption in kJ for various quantum computing environments. The plot employs coloured symbols where Secure/Efficient (1) is coloured differently from Insecure/Inefficient (0) [21]. The plot shows no consistent correlation between execution time and energy consumption and therefore these ideas can point towards a decentralized relationship, indicating that both time and energy are factors that can contribute to the security and improvements in the systems.

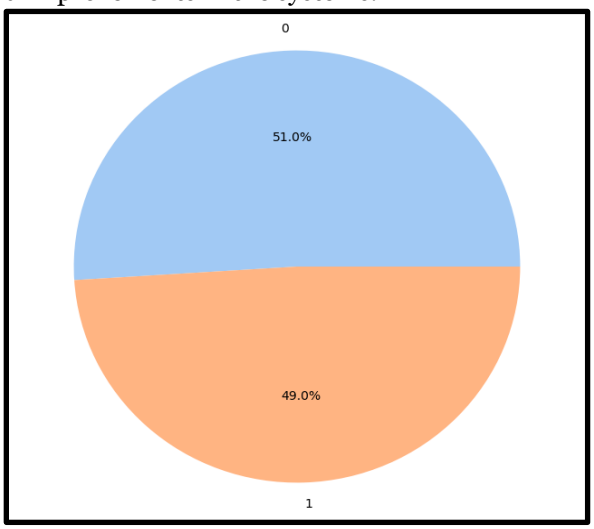


Fig 11: Pie Chart of Algorithm Type Distribution

This pie chart shows the proportion of different algorithms in the corpus of texts. There are two categories of algorithms in the dataset, categorized as 0 and 1 contributing to 51% and 49% respectively. This near-equal division means that the distribution of algorithm types is fair, thereby enabling one to properly compare how these different algorithms alter the security and efficiency of quantum computing.

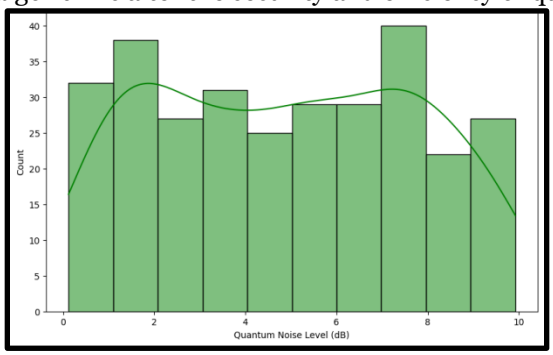


Fig 12: Histogram of Quantum Noise Level

This histogram reflects the distribution of Quantum Noise Levels (dB) within the dataset. The range is from about 0.12 dB to 9.92 dB, while most of them fall around 2 dB and 8 dB. Besides, there is also a KDE

line in the histogram, which reflects the pattern of variation underneath [22]. In this case, the variability in quantum noise, most likely impacting system performance, could be identified.

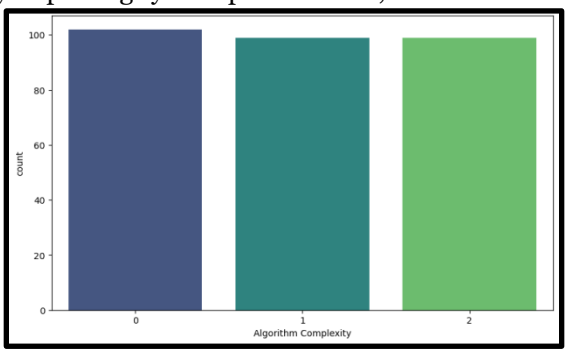


Fig 13: Bar Plot of Algorithm Complexity Distribution

The bar plot below shows the frequency of occurrence of the Algorithm Complexity levels in the chosen dataset. The complexity is divided into three categories: low, middle, and high, with each category totalling approximately one-third of the sample [23]. It is possible to have a more detailed comparison of how algorithm complexity impacts quantum computing results in terms of security and efficiency, thanks to this forward and backward approach.

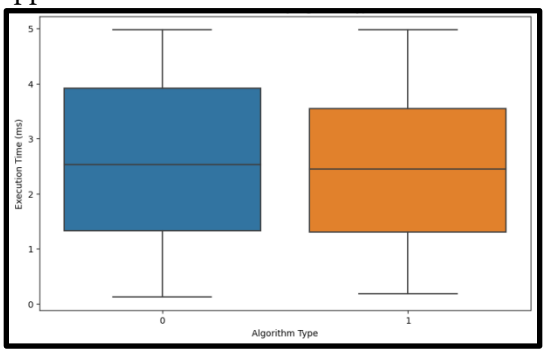


Fig 14: Execution Time by Algorithm Type

This box plot compares the dispersion of Execution Time (ms) for two Algorithm Types. Two kinds of algorithms illustrate comparable values of the median of the execution times but slightly different dispersion of the execution times. It also draws the possibility of outliers within a set as part of the plot. This comparison is important in establishing how the various algorithm types can positively impact the temporal complexity of quantum computing processes.

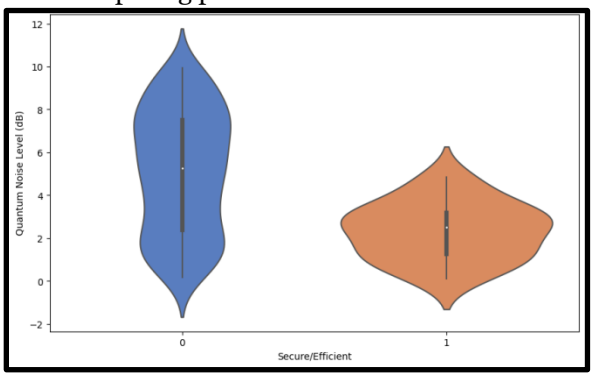


Fig 15: Quantum Noise Level by Secure/Efficient

The violin plot represents Quantum noise levels in dB concerning Secure/Efficient (1) and Insecure/Inefficient (0). The plot suggests that it is less variable in secure than in insecure settings, with more points clustered around the centre of the distribution. This means that lower quantum noise levels can be associated with greater security and efficiency of quantum computing systems, which is especially important when developing quantum algorithms.

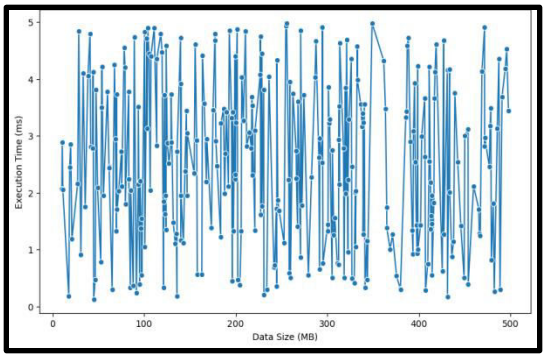


Fig 16: Execution Time vs Data Size

This line plot shows the quantum computing setups' Execution Time (ms) different from Data Size (MB). The graph also does not show any clear pattern of increase or decrease showing that the execution time does not increase in direct proportion to the size of the data input. However, there are variations in the duration of execution with the size of data as shown in the result tables above. This implies that there might be other variables like algorithmic intricacy, and noise levels that might also influence execution time greatly.

Logistic Regression Accuracy: 91.1
Classification Report for Logistic Regression:

	precision	recall	f1-score	support
0	0.98	0.93	0.95	86
1	0.25	0.50	0.33	4
accuracy			0.91	90
macro avg	0.61	0.72	0.64	90
weighted avg	0.94	0.91	0.92	90

Fig 17: Accuracy and Classification report of LR model

This figure represents the Logistic Regression model's accuracy and the classification report. The accuracy of this is 91.1%. From the classification report, for class 0, the precision is 0.98 and recall is 0.93, while for class 1, the precision falls to 0.25, showing that classifying a secure/efficient setup is hard.

Decision Tree Accuracy: 100.0
Classification Report for Decision Tree:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	86
1	1.00	1.00	1.00	4
accuracy			1.00	90
macro avg	1.00	1.00	1.00	90
weighted avg	1.00	1.00	1.00	90

Fig 18: DT model classification and accuracy score

The above figure shows the accuracy and classification report of the DT model. The accuracy score for this model is 100%, with precision and recall equal to 1.00 in all classes [24]. Thus, a Decision Tree model perfectly classified both secure/efficient and insecure/inefficient quantum computing setups, being the most effective model in this analysis.

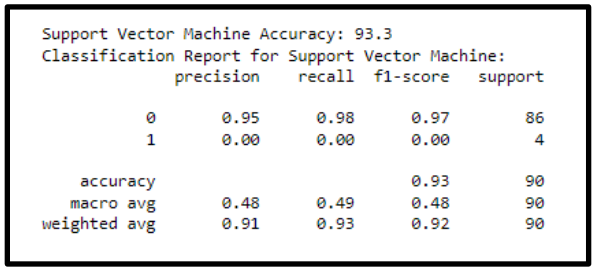
Random Forest Accuracy: 97.8
Classification Report for Random Forest:

	precision	recall	f1-score	support
0	0.98	1.00	0.99	86
1	1.00	0.50	0.67	4
accuracy			0.98	90
macro avg	0.99	0.75	0.83	90
weighted avg	0.98	0.98	0.97	90

Fig 19: RF model accuracy score and classification report

This figure shows that the RF model reached 97.8%, whereas the class classification report showed class 0 with a precision of 0.98 and class 1 of 1.00, with recall for class 1 being 0.50. Also, there are some

misclassification of the secure/efficient setups, but generally speaking, the model performs robustly with a high accuracy rate.

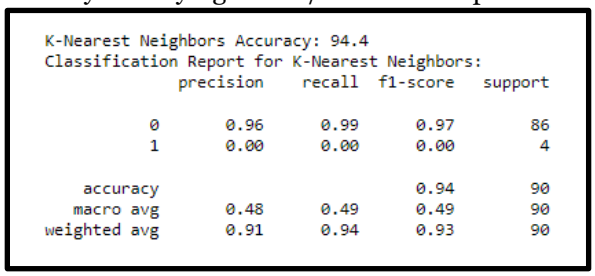


Support Vector Machine Accuracy: 93.3
Classification Report for Support Vector Machine:

	precision	recall	f1-score	support
0	0.95	0.98	0.97	86
1	0.00	0.00	0.00	4
accuracy			0.93	90
macro avg	0.48	0.49	0.48	90
weighted avg	0.91	0.93	0.92	90

Fig 20: Accuracy and Classification report of SVM model

The accuracy along with the classification report of the SVM model is shown in the above figure. This figure produced an accuracy of 93.3%, where it had a precision in class 0 of 0.95 and a recall of 0.98. On the other hand, the model fails miserably in class 1, with a precision of 0.00 and recall of 0.00, which implies complete failure in correctly classifying secure/efficient setups.



K-Nearest Neighbors Accuracy: 94.4
Classification Report for K-Nearest Neighbors:

	precision	recall	f1-score	support
0	0.96	0.99	0.97	86
1	0.00	0.00	0.00	4
accuracy			0.94	90
macro avg	0.48	0.49	0.49	90
weighted avg	0.91	0.94	0.93	90

Fig 21: Accuracy and Classification report of KNN model

The K-Nearest Neighbors model works with an accuracy of 94.4% shown in the above figure. The classification report indicates that class 0 has a precision of 0.96 and a recall of 0.99. Here, it fails with class 1 in which its precision and recall amount to 0.00, thereby failing to discover safe or efficient setups.

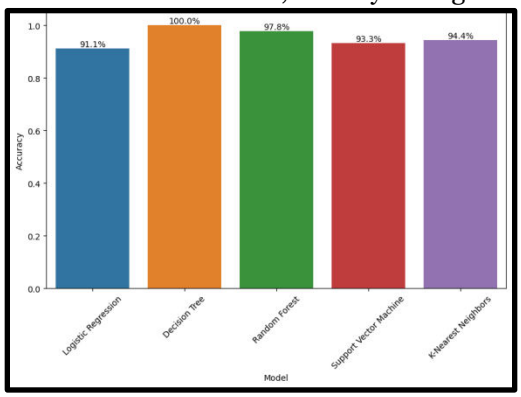


Fig 22: Model Accuracy Comparison

This figure compares all the accuracy scores of the analyzed models. The Decision Tree model stands first with 100% accuracy, whereas the Random Forest model's accuracy is 97%. 8%, K-Nearest Neighbors at 94. 4%, Naive Bayes at 92. 3% while the others are K Nearest Neighbor at 71. 1% [25]. By comparing the performance of the Decision Tree model to other models, this paper has been able to demonstrate that the Decision Tree model outperforms other models in classifying quantum computing setups.

```
encrypted_text = encrypt_data(sample_text, key)
print(f"Encrypted: {encrypted_text}")

# Decrypt the sample text
decrypted_text = decrypt_data(encrypted_text, key)
print(f"Decrypted: {decrypted_text}")
```

```
Encrypted: Bv5LfhJHQjNp5490H0SksQ==BFrfsH0frvg3aF4HcVCYmtFs4YY1Vv+THRDKfBEIx/sgu+liVLeuPXDy24/sjCxxRGP3RGhDq76tfNYup/XVV2w==
Decrypted: This is a secure message using quantum-resistant encryption.
```

```
[26]: key = get_random_bytes(16)  # AES-128
print(f"Encryption Key: {key.hex()}")
```

```
Encryption Key: 725f37836655044ba8688e3e7dc314f26
```

Fig 23: Encryption and Decryption using AES

The figure demonstrates the encryption and decryption of a secure message using AES-128 encryption. It showcases the hexadecimal format of the encryption key generated, illustrating the secure process.

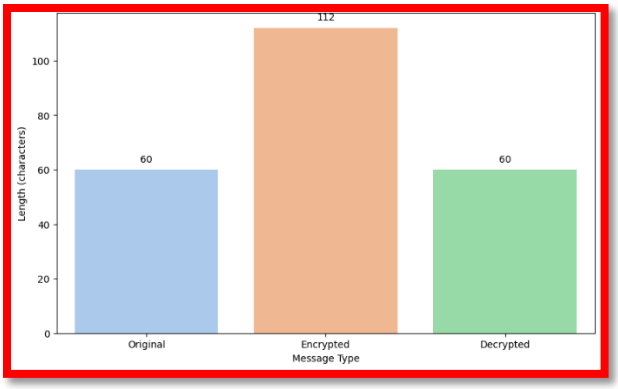


Fig 24: Comparison of Message Lengths

The figure compares the lengths of the original, encrypted, and decrypted messages. It shows that the encryption process significantly increases the message length due to added cryptographic data, while the decrypted message restores the original length, confirming the integrity of the encryption and decryption processes.

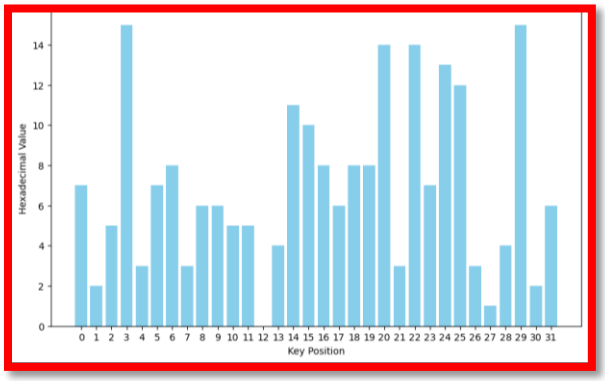


Fig 25: Visualization of Encryption Key Distribution

The distribution of the encryption key's hexadecimal values across different key positions which is shown in the above figure. It demonstrates the randomness and variation in key values, highlighting the secure nature of the AES encryption process used.

2. Discussion

In this analysis, predictive models are used to improve the classification of quantum computational environments with regards to secure or efficient. These algorithms evaluate features that include quantum noise levels, execution time, and data size, making it easier to detect patterns and correlations that may not be apparent in conventional approaches. Using Decision Trees and Random Forests models, the study is expected to enhance the classification procedure and provide more accurate measures of quantum computing security and performance, which are essential in making efficient quantum systems and cryptographic algorithms.

Model	**Accuracy (%)**	**Precision (Class 1)**	**Recall (Class 1)**
Logistic Regression	91.1	25.0%	50.0%
Decision Tree	100.0	100.0%	100.0%

Random Forest	97.8	100.0%	50.0%
Support Vector Machine	93.3	0.0%	0.0%
K-Nearest Neighbors	94.4	0.0%	0.0%

Table 2: Performance Comparison of Machine Learning Models

V. CONCLUSION AND RECOMMENDATION

Quantum computing becomes a potent threat to both cryptographic security and algorithm efficiency, as proved in this study. The overall findings show that the Decision Tree model is the most precise model among the four Machine Learning models since it achieved 100 per cent accuracy in differentiating between secure/ efficient setups and insecure/inefficient ones. However, other models like Random Forest and K-NN showed good performance as well. It was recommended that further research should try to enhance the effectiveness of such models, particularly where the underlying data is unevenly distributed. However, the idea of post-quantum cryptography needs to be investigated further, as well as look for more efficient algorithms to counteract the risks posed by quantum computing. It may be required to assess and improve cryptographic strategies and methods as the technology of quantum processors evolves.

REFERENCES

[1] Ajala, O.A., Arinze, C.A., Ofodile, O.C., Okoye, C.C. and Daraojimba, A.I., 2024. Exploring and reviewing the potential of quantum computing in enhancing cybersecurity encryption methods.
[2] Azhari, R. and Salsabila, A.N., 2024. Analyzing the Impact of Quantum Computing on Current Encryption Techniques. IAIC Transactions on Sustainable Digital Innovation (ITSDI), 5(2), pp.148-157.
[3] Coccia, M., Roshani, S. and Mosleh, M., 2022. Evolution of quantum computing: Theoretical and innovation management implications for emerging quantum industry. IEEE Transactions on Engineering Management, 71, pp.2270-2280.
[4] Radanliev, P., 2024. Artificial intelligence and quantum cryptography. Journal of Analytical Science and Technology, 15(1), p.4.
[5] Bova, F., Goldfarb, A. and Melko, R.G., 2021. Commercial applications of quantum computing. EPJ quantum technology, 8(1), p.2.
[6] Pal, S., 2023. Quantum computing and algorithms: an exploration of the quantum frontier in data analytics and computational intelligence. Juni Khyat, 13(10), pp.145-154.
[7] Kumar, M., 2022. Post-quantum cryptography Algorithm's standardization and performance analysis. Array, 15, p.100242.
[8] Kappert, N., Karger, E. and Kureljusic, M., 2021, July. Quantum Computing-The Impending End for the Blockchain?. In PACIS (p. 114).
[9] Khanal, B., Orduz, J., Rivas, P. and Baker, E., 2023. Supercomputing leverages quantum machine learning and Grover's algorithm. The Journal of Supercomputing, 79(6), pp.6918-6940.
[10] Akbar, M.A., Khan, A.A. and Hyrynsalmi, S., 2024. Role of quantum computing in shaping the future of 6 G technology. Information and Software Technology, 170, p.107454.
[11] Sabani, M.E., Savvas, I.K., Poulakis, D., Garani, G. and Makris, G.C., 2023. Evaluation and comparison of lattice-based cryptosystems for a secure quantum computing era. Electronics, 12(12), p.2643.
[12] How, M.L. and Cheah, S.M., 2023. Business Renaissance: Opportunities and challenges at the dawn of the Quantum Computing Era. Businesses, 3(4), pp.585-605.
[13] Kaleem, M., Mushtaq, M.A., Jamil, U., Ramay, S.A., Khan, T.A., Patel, S., Zahidy, R. and Hussain, S.K., 2024. New Efficient Cryptographic Techniques For Cloud Computing Security. Migration Letters, 21(S11), pp.13-28.
[14] Sáez-Ortuño, L., Huertas-Garcia, R., Forgas-Coll, S., Sánchez-García, J. and Puertas-Prats, E., 2024. Quantum computing for market research. Journal of Innovation & Knowledge, 9(3), p.100510.

[15] Ji, X., Wang, B., Hu, F., Wang, C. and Zhang, H., 2021. New advanced computing architecture for cryptography design and analysis by D-Wave quantum annealer. Tsinghua Science and Technology, 27(4), pp.751-759.
[16] Tandel, P.H. and Nasrıwala, J.V., 2022. Post-quantum cryptography: A solution to quantum computing on security approaches. In Pervasive Computing and Social Networking: Proceedings of ICPCSN 2021 (pp. 605-617). Springer Singapore.
[17] Nofer, M., Bauer, K., Hinz, O., van der Aalst, W. and Weinhardt, C., 2023. Quantum Computing. Business & Information Systems Engineering, 65(4), pp.361-367.

[18] Shamshad, S., Riaz, F., Riaz, R., Rizvi, S.S. and Abdulla, S., 2022. An enhanced architecture to resolve public-key cryptographic issues in the internet of things (IoT), employing quantum computing supremacy. Sensors, 22(21), p.8151.
[19] Niraula, T., Pokharel, A., Phuyal, A., Palikhel, P. and Pokharel, M., 2022. Quantum computers' threat on current cryptographic measures and possible solutions. Int. J. Wirel. Microw. Technol, 12(5), pp.10-20.
[20] Abd El-Aziz, R.M., Taloba, A.I. and Alghamdi, F.A., 2022. Quantum computing optimization technique for iot platform using modified deep residual approach. Alexandria Engineering Journal, 61(12), pp.12497-12509.
[21] Ur Rasool, R., Ahmad, H.F., Rafique, W., Qayyum, A., Qadir, J. and Anwar, Z., 2023. Quantum computing for healthcare: A review. Future Internet, 15(3), p.94.
[22] Hussain, S., Neupane, Y., Wang, W.L., Ibrahim, N., Khan, S.U.R. and Kareem, A., 2022, June. Empirical Investigation of Quantum Computing on Solving Complex Problems. In International Conference on Agile Software Development (pp. 222-230). Cham: Springer Nature Switzerland.
[23] Awan, U., Hannola, L., Tandon, A., Goyal, R.K. and Dhir, A., 2022. Quantum computing challenges in the software industry. A fuzzy AHP-based approach. Information and Software Technology, 147, p.106896.
[24] Yalcin, H., Daim, T., Moughari, M.M. and Mermoud, A., 2024. Supercomputers and quantum computing on the axis of cyber security. Technology in Society, 77, p.102556.
[25] Kumar, A., Bhatia, S., Kaushik, K., Gandhi, S.M., Devi, S.G., Diego, A.D.J. and Mashat, A., 2021. Survey of promising technologies for quantum drones and networks. Ieee Access, 9, pp.125868-125911.