GMR
The Original

QUANTUM DRIVEN BAYESIAN OPTIMIZATION FOR LIVER CIRRHOSIS STAGE PREDICTION

Ashok Kumar Panigrahi[1], Chittaranjan Mallick[2], and Kalyan Kumar Jena[3*]

[1]Faculty of Computer Application and Sciences, Biju Patnaik University of Technology, Rourkela, 769015, Odisha, India, Email: ashokpanigrahimath@gmail.com, http://orcid.org/0009-0003-0664-5010
[2]Department of Mathematics, Parala Maharaja Engineering College, Berhampur, Odisha, India,
Email: cmallick75@gmail.com, http://orcid.org/ 0000-0002-4165-652X
[3]Department of Computer Science and Engineering, Parala Maharaja Engineering College, Berhampur, Odisha, India,
Email: kalyankumarjena1@gmail.com, http://orcid.org/ 0000-0002-2476-5644

Abstract
There are several ways to predict Liver cirrhosis stage (LCS) such as quantum computing (QC), machine learning (ML) and deep learning (DL), etc. This paper is dedicated to the use of Quantum Machine Learning (QML) to predict the conditions of LCS being either mild (ML), moderate (MD) or severe (SV). In particular, it will focus on using the Quantum Gradient Boosting (QGB) model, the Quantum Support Vector Machine (QSM) model and the Hybrid Quantum-Classical Model (HQM) to predict LCS.In addition, all the models are compared using metrics such as classification accuracy (A), precision (P), recall (R), F1 Score (F1S), training time (TT), memory usage (M) and model size (MS) by analyzing 80/20, 70/30 and 60/40 training/testing ratios prior and post the application of hyperparameter Bayesian optimization (HPO) tuning. To compare model performance metrics, the receiver operating characteristic (ROC) curve will be plotted based on true positive rate (TPR) and false positive rate (FPR) for each model and visualized using boxplots and heatmaps based on each metric. The findings suggest that HQM performs better than its competitors across all measures of A, P, R, F1S, and TT with average scores (95.53, 95.11, 95.25, 95.29 and 5.60) before BO, and (97.63, 97.52, 97.58, 97.55 and 5.46) after, therefore it is clear that HQM outperforms the competition. However, QSM performs better in terms of of M and MS with values 1275.19 and 89.31, and 1270.47 and 81.31 accordingly.

Keywords: Quantum Machine Learning, Liver Cirrhosis, Bayesian Optimization, Classification Accuracy, Precision, Recall, F1 Score, Training Time, Memory Usage, Model Size.

1. INTRODUCTION

The liver is a crucial organ that performs a key role in digestion, detoxification, metabolism and many other crucial functions. Liver cirrhosis [1-10] refers to An advanced stage of liver fibrosis resulting from various liver disorders including long term alcohol consumption, viral infections, genetic disorders, etc. Cirrhosis occurs when the liver cells are damaged over time and its tissue becomes scarred and hardened. The different symptoms include fatigue, jaundice, Edema, bleeding, itchy skin, loss of appetite, weight Loss, confusion, etc. So, it is very much important for early LCS prediction for early diagnosis, tailored treatment plans and monitoring disease progression otherwise it may lead to liver cancer, portal hypertension and liver failure. The diagnosis of liver cirrhosis includes medications, lifestyle changes, liver transplantation, etc. However, some preventive measures include avoiding excessive alcohol, healthy diet plan, weight management, regular follow up and monitoring of liver function.

The LCS [11-20] can be tracked using a variety of methods, including Machine Learning (ML), Deep Learning (DL), Quantum Machine Learning (QML), Federated Learning(FL), etc. Various ML-based models can be used for such predictions including Support Vector Machines (SVMs), Random Forests (RFs), Decision Trees (DTs), Naive Bayes (NB), and K-Nearest Neighbors (KNN), among others. Numerous DL models can also be employed for this purpose: Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Autoencoders, Deep Reinforcement Learning (DRL), etc. Other approaches to LCS tracking such as QML techniques can utilize Quantum-based Systems (QSM), Quantum Neural Network (QNN), Quantum-based Genetic Programming (QGB) and Hierarchical Quantum Mechanics (HQM). Among other models of FL, Federated Averaging, Federated Stochastic Gradient Descent, Federated Multi-Task Learning etc. are each viable solutions to tracking metrics at scale. Genetic and swarm intelligence algorithms are also potential solutions to the tracking problem.

Here, it emphasizes QML's advantages compared with other methods, including its ability to efficiently handle high-dimensional data; improve the extraction of features and the classification, provide faster computation in general; and handle noisy data better than other options do. The potential of QML [21-27] Models in predicting the LCS Conditions is very high. QML Provides better performance and faster computation while being able to handle more complex and larger amounts of data than could be done by any other means. The use of quantum superposition and entanglement allows

for the handling of complexity in these predictions through the ability to examine many different possibilities at the same time, thereby enhancing the prediction process on LCS. QML has the capability of detecting fine patterns in data that may not be detected by traditional ML methods. QML will also be able to detect potential problems with greater accuracy and in real-time than could be done by classical ML methods, thus allowing for an improvement in the accuracy of LCS predictions.

The primary goal of using this approach for LCS classification is to increase the efficiency, accuracy, and scalability of health monitoring systems for LCS, while also taking advantage of quantum computing (QC). Classic ML typically cannot cope with the volume and dimensionality of real-world data, but QC based solutions can process data in parallel, and are effective at capturing complex and/or non-linear relationships in the data. QC techniques are able to accommodate uncertainty and/or noise in the data, adapt to dynamic operational environments, and enhance feature selection, which will help to improve overall A. The QML approach will be used in this work to predict LCS with the ML, MD, and SV approaches. LCS will be predicted using QGB, QSM, and HQM models. All three models will be compared based on CA, P, R, F1S, TT, M, and MS over multiple types of test runs. The results are then visualized with a ROC curve, boxplot, and heatmap. Below is a summary of the major contributions made in this study:

- A QML-based methodology has been developed to predict LCS as ML, MD, or SV.
- QGB, QSM, and HQM models have been utilized as tracking tools for LCS.
- The models were compared based on A, P, R, F1S, TT, M, and MS using TTR of 80:20, 70:30, and 60:40, both before and after applying BO.
- BO was employed to enhance and optimize the performance of the models.
- ROC, boxplot, and heatmap were used to visualize and analyze the results.
- HQM outperforms other models in terms of A, P, R, F1S, and TT with average values of 94.1, 94.0, 94.1, 94.2, and 4.49 before BO, and 96.9, 96.8, 96.8, 96.8, and 4.35 after BO.
- QSM performs better in terms of M and MS with values of 1274.41 and 1278.36 before BO, and 1275.54 and 1270.36 after BO.
- The implementation was carried out using Jupyter Notebook 7.2.1.

The layout of rest of the paper is outlined below. Sections 2 to 7 cover the literature review, problem description, methodology, results and discussion, statistical analysis and conclusion respectively.

2. RELATED WORKS

Many research studies have been conducted concerning QML in medical scenarios [1-27]. Some of the important works are discussed below.

Li et al. [1] presented a DL approach for predicting drug-induced liver injury (DILI) by constructing a molecular representation framework that embeds electronic properties onto the molecular surface using manifold techniques. Local electronic properties of the molecular surface are mapped onto a lower-dimensional manifold, and the embedding is transformed into a matrix form to serve as input for a DL model. The model is trained using a dataset and validated through cross-validation (CV) techniques. Ayalew et al. [2] introduced an approach for the rapid detection and classification of sarcoidosis using patient chest X-ray data. Advanced models such as Inception and Residual Network-50, along with a handcrafted DL method and a QSM classifier, were employed for disease identification. The framework integrates a CNN and histogram of oriented gradients (HOG) technique to assist healthcare professionals in detecting sarcoidosis. To improve image quality, anisotropic diffusion filtering was applied to preserve edges, reduce noise, and enhance the images for better analysis. Thoufeeq et al. [3] investigated metabolic changes occurring when a normal liver progresses into cirrhosis and hepatocellular carcinoma (HCC). Significant differences were observed between the average fluorescence lifetime values at 350 nm and 450 nm emissions for cirrhosis and HCC samples when compared with normal liver tissue. The classification results generated using hierarchical cluster analysis (HCA) and ROC analysis based on fluorescence lifetime values successfully discriminated normal liver tissue from diseased tissues with an average sensitivity of 86.95% and specificity of 96.43%. Lusnig et al. [4] addressed diagnostic accuracy challenges through advanced QML techniques known for enhanced generalization capability, while also considering privacy concerns using privacy-preserving federated learning (FL). A hybrid quantum neural network (QNN) model utilizing real-world clinical data was proposed to accurately diagnose non-alcoholic liver steatosis. The model achieved 97% image classification accuracy, outperforming conventional approaches by 1.8%. Pomarico et al. [5] applied quantum-inspired algorithms for the classification of HCC tissues using microarray gene expression data. By utilizing previously identified genetic communities, the computational complexity associated with the number of qubits was reduced, enabling quantum-inspired methods to be efficiently executed on classical computing systems. Rahi et al. [7] introduced a DNN-based model with L2 regularization for predicting liver disease risk across five categories, namely healthy, low risk, moderate risk, high risk, and severe liver disease.

From the analysis of these approaches there were noted a number of relevant research gaps, which include - very little research has been done into the use of quantum algorithms for LCS prediction, there are many opportunities available to apply quantum feature mapping to large datasets but this method is not widely used; most of the quantum based models developed are not scalable; quantum computing and classical machine learning are rarely integrated; quantum computing has seen very few examples of how quantum techniques have been applied in real time applications; no agreed-upon

benchmark per se currently exists for QML applications; many quantum models lack the ability to be improved; there is not enough emphasis on the integration of QML into prediction problems. Therefore, there is clearly a need for the creation of better quantum methods to address these problems.

3. PROBLEM STATEMENT

Utilizing a QML-based method for predicting LCS can be considered as a classification task to determine whether a sample belongs to NR or non-NR categories. The dataset can be represented as $X = \{x_1, x_2, ..., x_n\}$, where each xi denotes the feature vector of a sample containing sensor measurements such as temperature, pressure, vibration, etc. Each sample is associated with a label $y_i \in \{0,1\}$, where 0 represents NR and 1 represents non-NR classes such as ML, MD, and SV.

4. METHODOLOGY

The techniques highlighted in this study can be viewed in Figure 1. This dataset (D) is obtained through a source [28]. D is pre-processed with imputation (using feature mean), normalization of the data (using Z score), and then D is reduced using PCA. Once this is done the QML models are implemented on D to evaluate the performance based on the metrics given in Table 1 for three different TTR values (80/20, 70/30 and 60/40). ROC (plot measures of TPR/FPR) are plotted for each model and box plots and heat maps are created to visualise the results from using the metrics in Table 1. Subsequently, LCS can be grouped into general classes: ML , MD , and SV.

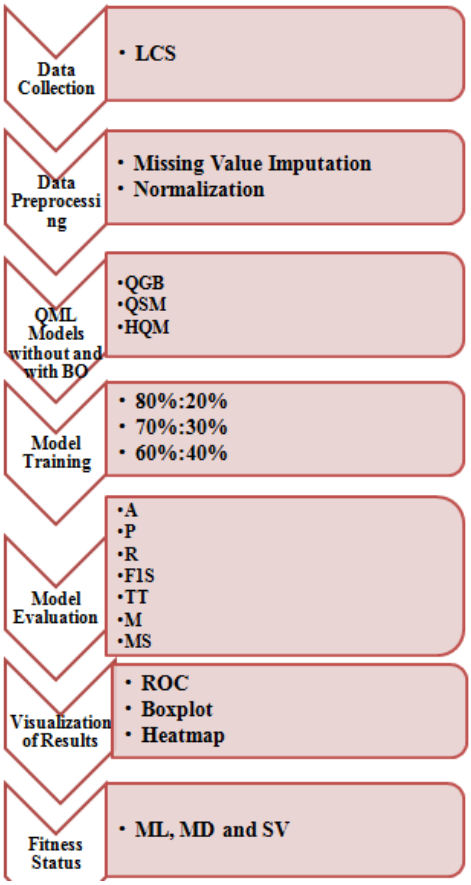


Figure 1. Methodology

4.1 Proposed Approach

The first stage of the proposed approach examines the classification data that has been passively pre-processed (through feature mean imputation to correct for missing values and z-score normalization for data normalisation). After that, PCA is implemented in order to reduce the number of features. The features generated after performing this PCA will be equal to the number of quantum features that were used in the quantum circuits when transforming from classical feature space to quantum state space. The classical machine learning methods (QGB, QSM and HQM (Staking) models) will then be used to classify the quantum data generated from this process. The QGB, QSM and HQM algorithms are without and with BO are mentioned in Algorithms 1- 6 accordingly.

Algorithm 1: QGB without BO

1. Input: D, number of qubits (NQ), Test Size
2. Output: A, P, R, F1S, TT, M, MS
3. Load D with feature set X and labels Y
4. For each feature f in X:

4.1 Apply mean value imputation to replace missing values
4.2 Normalize f using Z-score normalization

Gene

Algorithm 2: QSM without BO

1. Input: D, NQ, Test Size
2. Output: A, P, R, F1S, TT, M, MS
3. Load D with feature set X and labels Y
4. For each feature f in X:

4.1 Apply mean value imputation

4.2 Normalize using Z-score normalization

5. Combine all normalized features into Normalized_X
6. Apply PCA with ND = NQ to obtain Fscaled
7. Apply quantum feature map to obtain QF
8. Split data into training and testing sets
9. Train SVM classifier (linear kernel) on training data
10. Predict labels on test data

Compute performance metrics A, P, R, F1S, TT, M, MS

Algorithm 3: HQM without BO

1. Input: D, NQ, Test Size
2. Output: A, P, R, F1S, TT, M, MS
3. Load dataset D with X and Y
4. For each feature f in X:

4.1 Apply mean value imputation

4.2 Normalize using Z-score normalization

5. Combine normalized features into Normalized_X
6. Apply PCA with ND = NQ to obtain Fscaled
7. Apply quantum feature map to obtain QF
8. Split data into training and testing sets
9. Train Gradient Boosting classifier (CGB)
10. Train stacking-based hybrid model using base learners (GB + Quantum GB)
11. Fit hybrid classifier on training data
12. Predict labels on test data

Compute performance metrics A, P, R, F1S, TT, M, MS

Algorithm 4: QGB with BO

1. Input: D, NQ, Test Size, Hyperparameter range
2. Output: A, P, R, F1S, TT, F, MS
3. Load dataset D with X and Y
4. For each feature f in X:

4.1 Apply mean value imputation

Algorithm 5: QSM with BO

1. Input: D, NQ, Test Size, Hyperparameter range
2. Output: A, P, R, F1S, TT, F, MS
3. Load dataset D with X and Y
4. For each feature f in X:

4.1 Apply mean value imputation

4.2 Normalize using Z-score normalization

5. Combine normalized features into Normalized_X
6. Apply PCA with ND = NQ
7. Apply quantum feature map to obtain QF
8. Split dataset into training and testing sets
9. Define hyperparameter search space
10. Apply Bayesian Optimization for SVM
11. Train optimized SVM model
12. Predict labels on test set

Compute performance metrics A, P, R, F1S, TT, F, MS

Algorithm 6: HQM with BO

1. Input: D, NQ, Test Size, Hyperparameter range
2. Output: A, P, R, F1S, TT, F, MS
3. Load dataset D with X and Y
4. For each feature f in X:
5. Apply mean value imputation

4.2 Normalize using Z-score normalization

6. Combine normalized features into Normalized_X
7. Apply PCA with ND = NQ
8. Apply quantum feature map to obtain QF
9. Split dataset into training and testing sets
10. Define hyperparameter search space
11. Apply Bayesian Optimization for HQM
12. Train optimized HQM model
13. Train stacking classifier using optimized base models
14. Fit hybrid model on training data
15. Predict labels on test set
16. Compute performance metrics A, P, R, F1S, TT, F, MS

4.1.1 Input Data Representation

Let the dataset contain n samples, and each sample has m features. The dataset is represented as matrix X, and the output labels are stored in vector Y. X contains feature values of all samples, while Y contains class labels (0 or 1) for each sample.

4.1.2 Data Preprocessing

Handling Missing Values
Missing values in the dataset are replaced using the mean value of that feature.

4.1.2.1 Normalization
All features are normalized to bring them to a common scale. Normalized value (Xnorm) is mentioned in Equation (1)
.Xnorm = (X − mean) / standard deviation (1)

4.1.2.2 Dimensionality Reduction (PCA)
If the number of features is large, PCA is applied to reduce dimensions while keeping important information. It is mentioned in Equation (2).XPCA = PCA(Xnorm) (2)

4.1.2.3Quantum Feature Mapping
After PCA, each data point is converted into a quantum representation using a quantum feature map as mentioned in Equation (3).
Qi = MQ(XiPCA) (3)
Here, each classical data point is transformed into a quantum state using quantum gates such as RY rotation, CNOT, and Pauli-Z.

4.1.2.4 Train-Test Split
The dataset is divided into training and testing sets (20%, 30%, 40% test size). It is mentioned in Equation (4).
(Xtrain, Xtest, Ytrain, Ytest) = split(X, Y) (4)

4.1.2.5 QGB
In QGB, a Gradient Boosting classifier is applied to quantum-transformed data. The input data is first processed using a quantum feature mapping technique, and then it is used to train the model. QGB is trained using the training dataset (Xtrain, Ytrain). After training, the model predicts the output labels for unseen data is mentioned in Equation (5).
Ypred = QGB(Xtest) (5)
This model learns patterns from quantum-enhanced features to improve classification performance.

4.1.2.6 QSM
In QSM, a SVM is trained on quantum-transformed features. The quantum mapping helps to transform the data into a higher-dimensional space, making it easier to separate different classes. SVM is trained using Xtrain and Ytrain. After training, it predicts labels for test data as mentioned in Equation (6).
Ypred = SVM(Xtest) (6)
This method improves classification by using quantum feature space representation.

4.1.2.7 HQM
HQM is a hybrid model that combines two different learners:
- F1: Classical Gradient Boosting model
- F2: Quantum Gradient Boosting model

Both models independently learn from the same input data.

Prediction process:
P1 = F1(X) (7)→ prediction from classical model
P2 = F2(X) (8)→ prediction from quantum model
These outputs are then combined using a Logistic Regression model, which acts as a final decision maker.

Final prediction:
Pfinal = LogisticRegression(P1, P2) (9)
This hybrid approach improves accuracy by combining both classical and quantum learning strengths.

4.1.2.8 BO
Bayesian Optimization is used to automatically find the best hyperparameters for the model instead of manually trying different values. First, a set of possible hyperparameters (search space) is defined. The model is then trained and tested using different combinations of these parameters. Each combination is evaluated using a performance score calculated from validation data as mentioned in Equation (10).
F(HP) = average performance across validation folds (10)
The BO process then selects the best hyperparameter set that gives the highest performance as mentioned in Equation (11): HP* = best hyperparameters (11)

Finally, the model is trained using these optimal values:
BestModel = model with optimized hyperparameters from BO
This method improves performance while reducing the time needed for tuning.

4.1.3 Performance Metrics

A: It represents the proportion of correctly predicted samples out of the total test samples. It measures overall correctness of the model.
P : It indicates how many of the predicted positive samples are actually correct. It evaluates the reliability of positive predictions.
R : It measures how many actual positive samples are correctly identified by the model. It reflects the model's ability to detect positive cases.
F1S:It is the harmonic mean of precision and recall. It provides a balanced measure of model performance when both metrics are important.
TT: It is the total time taken by the model from the start of training to its completion. It reflects the computational efficiency of the model.
M : It represents the amount of RAM consumed during model execution, usually measured in megabytes.
MS: It indicates the storage size of the trained model, typically measured in kilobytes. It shows how compact or large the trained model is.

5 RESULTS AND DISCUSSION

Here, the D is taken from the source [28] for processing. The QML models are analyzed using the TTR such as 80%:20%, 70%:30% and 60%:40%. The performance is assessed based on parameters (as mentioned in equation (23)-(29)). The performance of these models are also visualized using ROC curve, Boxplot and Heatmap representation. The results of several methods by considering the TTR as 80%:20%, 70%:30% and 60%:40% by considering before and after BO are mentioned in Table 1-8 and Figure 2 – Figure 17.

Table 1: Results of several methods at 80/20 TTR before BO

Method	A	P	R	F1S	TT	M	MS
QGB	94.81	94.80	94.90	94.73	128.93	1286.30	179.50
QSM	91.40	91.48	91.35	90.82	116.3028	1277.80	90.58
HQM	96.65	96.60	96.61	96.53	5.67	2336.59	529.64

Table 2: Results of several methods at 80/20 TTR after BO

Method	A	P	R	F1S	TT	M	MS
QGB	99.03	98.92	99.02	98.98	120.0228	1276.22	173.38
QSM	96.71	96.84	96.78	96.20	111.2027	1271.73	82.44
HQM	99.70	99.63	99.73	99.64	5.53	2316.54	521.60

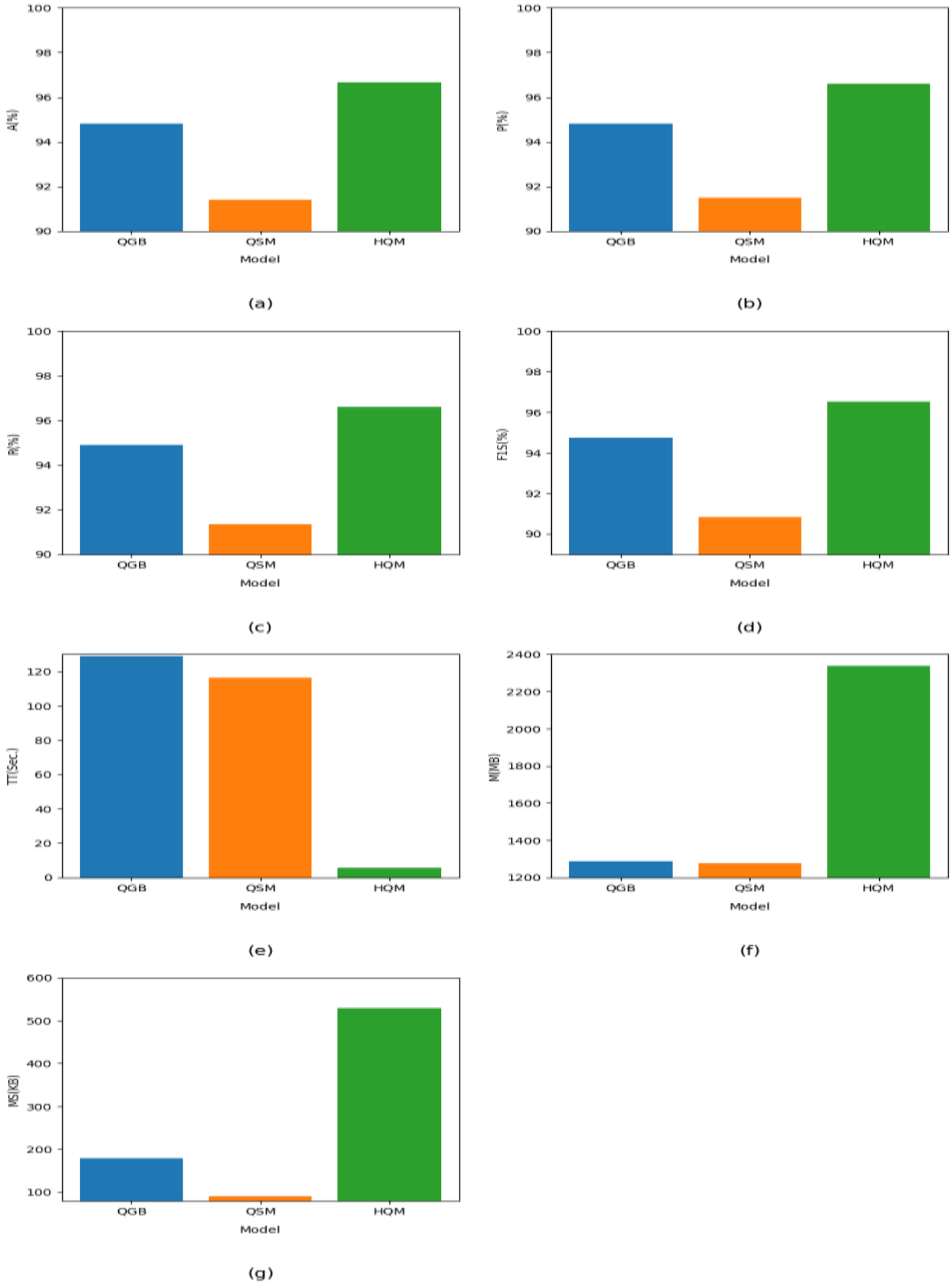
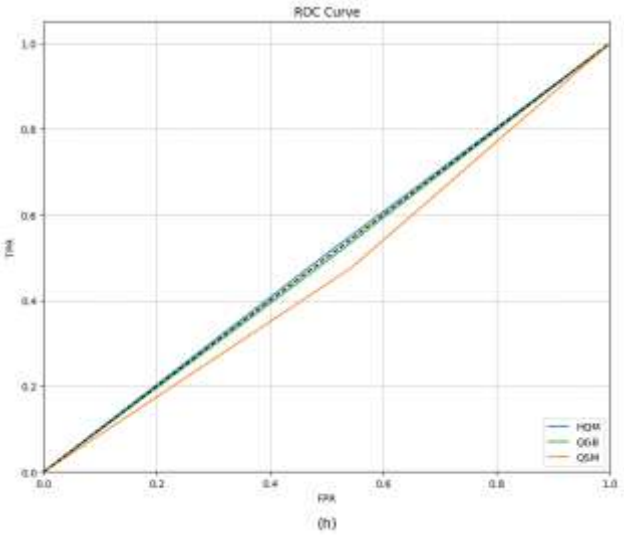


Figure 2. Results representation at 80/20 TTR before BO



(h)

Figure 3. ROC: 80/20 TTR before BO

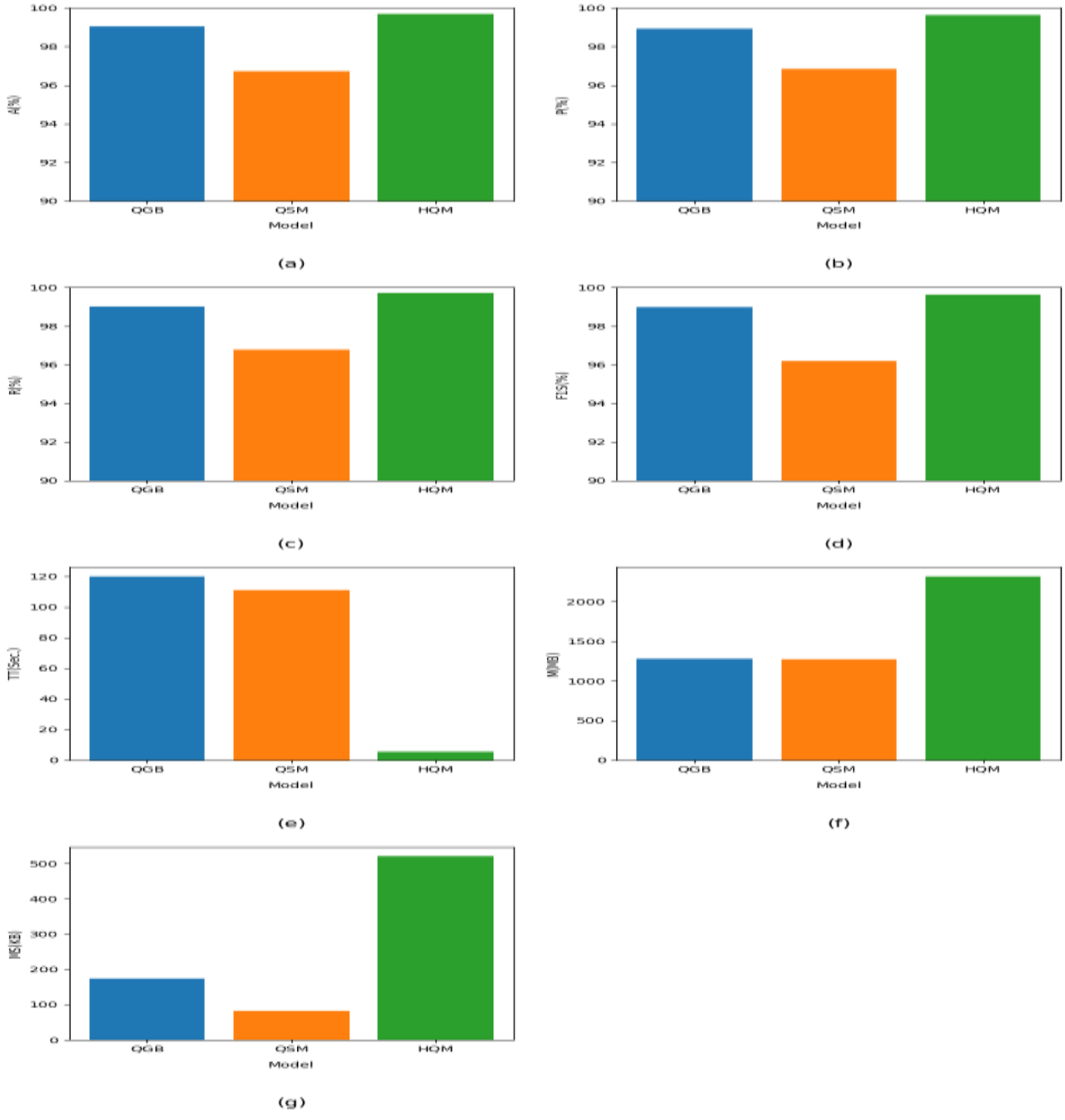


Figure 4. Results representation at 80/20 TTR after BO

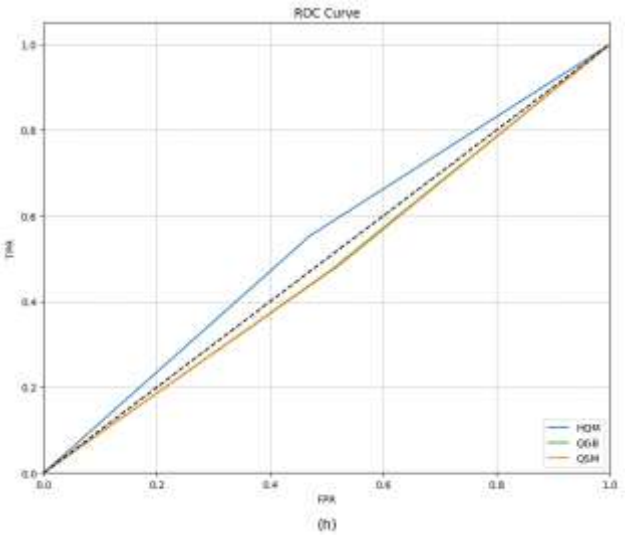


Figure 5. ROC: 80/20 TTR after BO

For the 80%:20% TTR (Tables 1 - 2 and Figure 2 - 5), the models QGB, QSM, and HQM are evaluated before and after BO. Before BO, QGB, QSM and HQM achieved the A of 94.81%, 91.40% and 96.65% accordingly. HQM achieved P of 96.60% and R of 96.61%, QGB also achieved the P of 94.80% and R of 94.90% while QSM has comparatively lower P of 91.48% and R of 91.35%. The highest F1S achieved for HQM of 96.53% followed by QGB of 94.73%, and QSM of 90.82%. HQM has faster TT with 5.67 sec., with QGB at 128.93 sec. and QSM at 116.3028 sec. HQM has the higher M of 2336.59 MB and MS of 529.64 KB, while QGB has 1286.30 MB and 179.50 KB, and QSM has 1277.80 MB and 90.58 KB.

After BO, HQM's A increased to 99.70%, QGB's to 99.03%, and QSM's to 96.71%. P for QGB to 98.92%, R to 99.02%, and F1S to 98.98%. HQM's P to 99.63%, R to 99.73%, and F1S to 99.64%, while QSM's P to 96.84%, R to 96.78%, and F1S to 96.20%. TT after BO is reduced to 120.0228 sec. for QGB, 111.2027 sec. for QSM, and 5.53 sec. for HQM. M decreased for HQM to 2316.54 MB, for QGB to 1276.22 MB, for QSM to 1271.73 MB. Similarly, MS for HQM, QGB and QSM reduced to 521.60 KB, 173.38 KB and 82.44 KB accordingly.

TPR/FPR before BO showed QGB with 0.76/0.00, QSM with 0.69/0.00, and HQM with 0.90/0.00. After BO, TPR improved to 0.78 for QGB, 0.70 for QSM, and 0.92 for HQM. The ROC curve is a key metric for evaluating the performance of classifiers, particularly with respect to their ability to distinguish between positive and negative classes. In the 80%:20% TTR scenario, the ROC curve is analyzed for the models HQM, QGB and QSM both before and after BO. All models exhibited a FPR of 0, indicating that none of the models misclassified negative instances as positive. This is reflected in the models' positions along the y-axis of the ROC curve. Before BO, the TPR for QGB, QSM, and HQM are 0.76, 0.69, and 0.90, respectively, with HQM showing higher TPRs, indicating that it performed better in capturing TP instances compared to QGB and QSM. The ROC curve for HQM appeared higher along the curve, while QGB and QSM are positioned comparatively lower, indicating their relative underperformance. After BO, all models showed an improvement in TPR, with QGB at 0.78, QSM at 0.70, and HQM at 0.92. This increase in TPR suggests that HQM is capable of identifying TP instances after optimization. So, HQM moved further up the ROC curve, reflecting enhanced classification performance, while QGB and QSM also showed comparative improvements.

Similarly, for the 70%:30% TTR (as mentioned in Tables 3- 4 and Figure 6-9), the models QGB, QSM, and HQM are evaluated before and after BO. For the 60%:40% TTR (as mentioned in Tables 5 - 6 and Figure 10 - 13), the models QGB, QSM, and HQM are evaluated before and after BO.

Table 3: Results of several methods at 70/30 TTR before BO

Method	A	P	R	F1S	TT	M	MS
QGB	93.42	93.43	93.54	93.43	127.62	1284.71	177.99
QSM	89.88	90.00	89.83	89.38	115.807	1274.92	89.62
HQM	95.52	95.43	95.52	95.57	5.62	2334.33	527.89

Table 4: Results of several methods at 70/30 TTR after BO

Method	A	P	R	F1S	TT	M	MS
QGB	95.92	95.23	95.83	95.84	117.90	1274.87	172.24
QSM	93.42	93.59	93.47	92.99	109.92	1270.36	81.29
HQM	97.60	97.52	97.62	97.52	5.48	2305.55	519.56

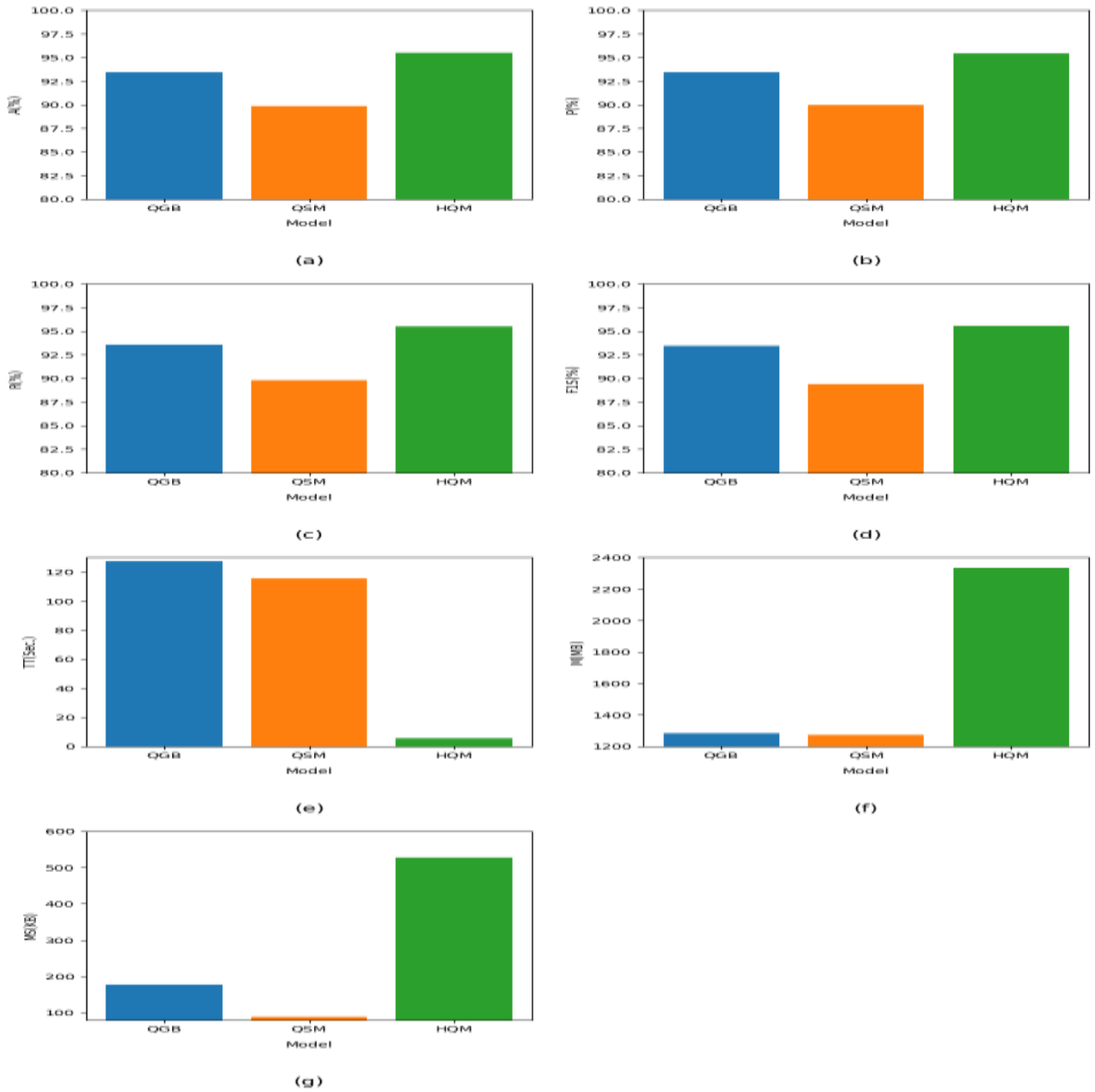


Figure 6. Results representation at 70/30 TTR before BO

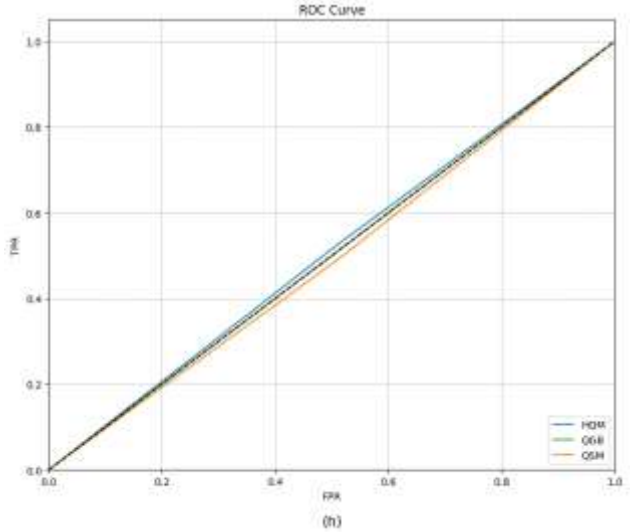


Figure 7. ROC: 70/30 TTR before BO

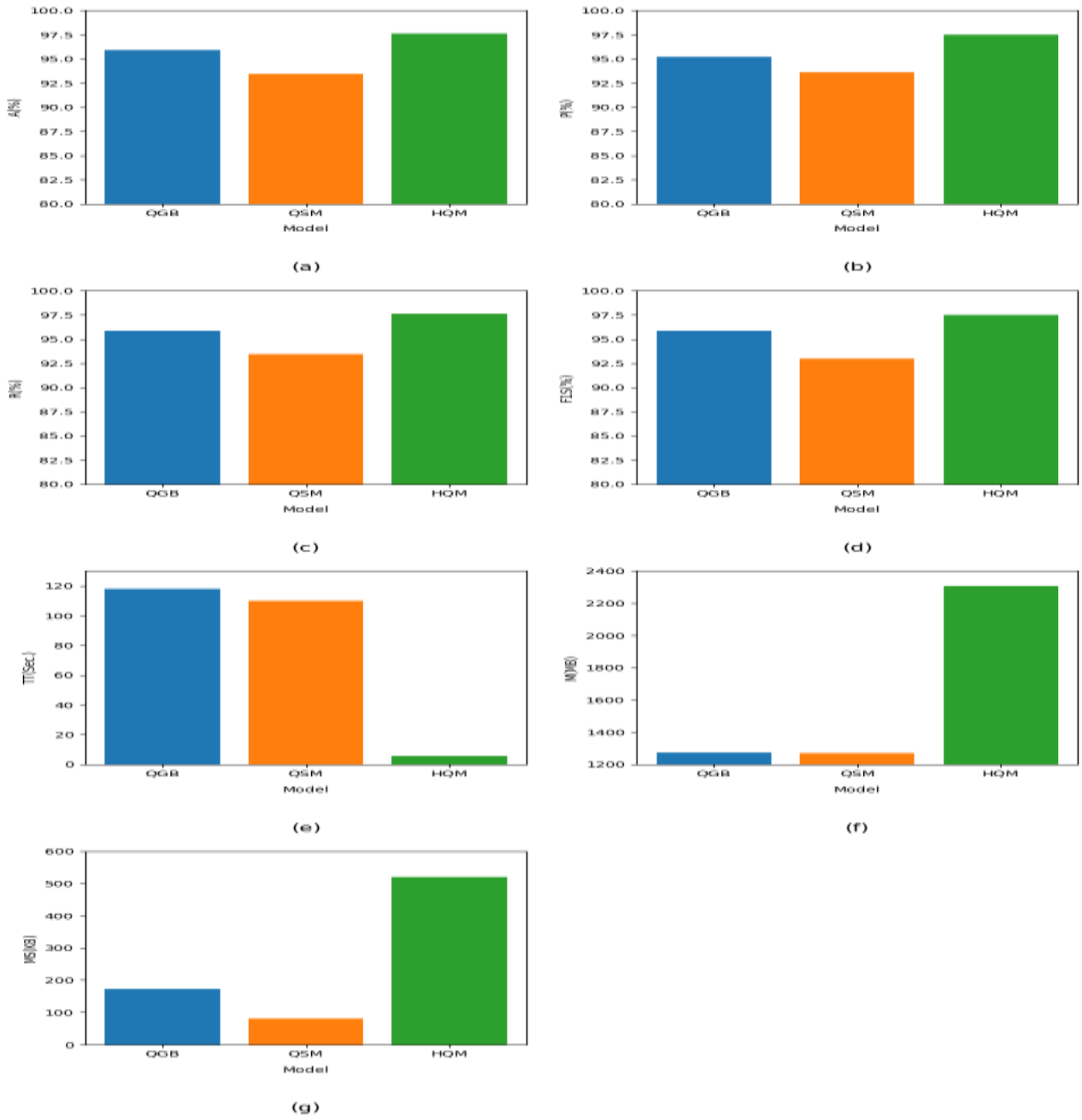


Figure 8. ROC: 70/30 TTR after BO

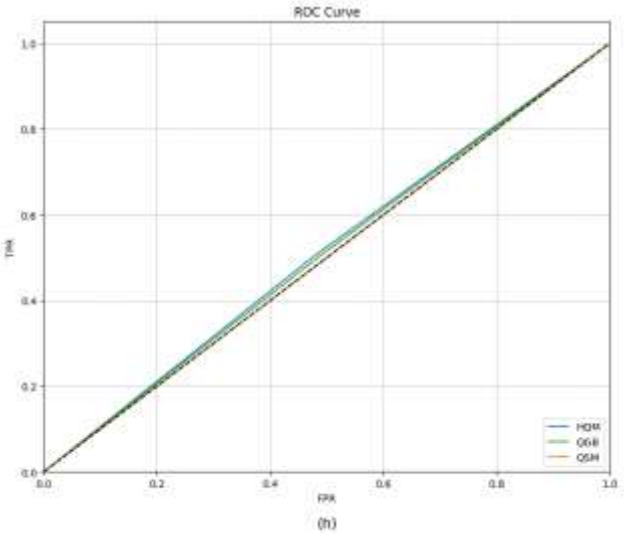


Figure 9. ROC: 70/30 TTR after BO

Table 5: Results of several methods at 60/40 TTR before BO

Method	A	P	R	F1S	TT	M	MS
QGB	92.22	92.70	92.42	92.40	126.42	1282.70	176.65
QSM	88.39	88.39	88.99	87.46	114.40	1273.84	88.74
HQM	93.42	93.29	93.61	93.78	5.50	2333.23	526.34

Table 6: Results of several methods at 60/40 TTR after BO

Method	A	P	R	F1S	TT	M	MS
QGB	93.70	93.64	93.63	93.64	116.26	1273.75	171.84
QSM	92.23	92.39	92.30	91.72	108.72	1270.32	81.20
HQM	96.60	96.42	96.40	96.50	5.37	2305.50	518.54

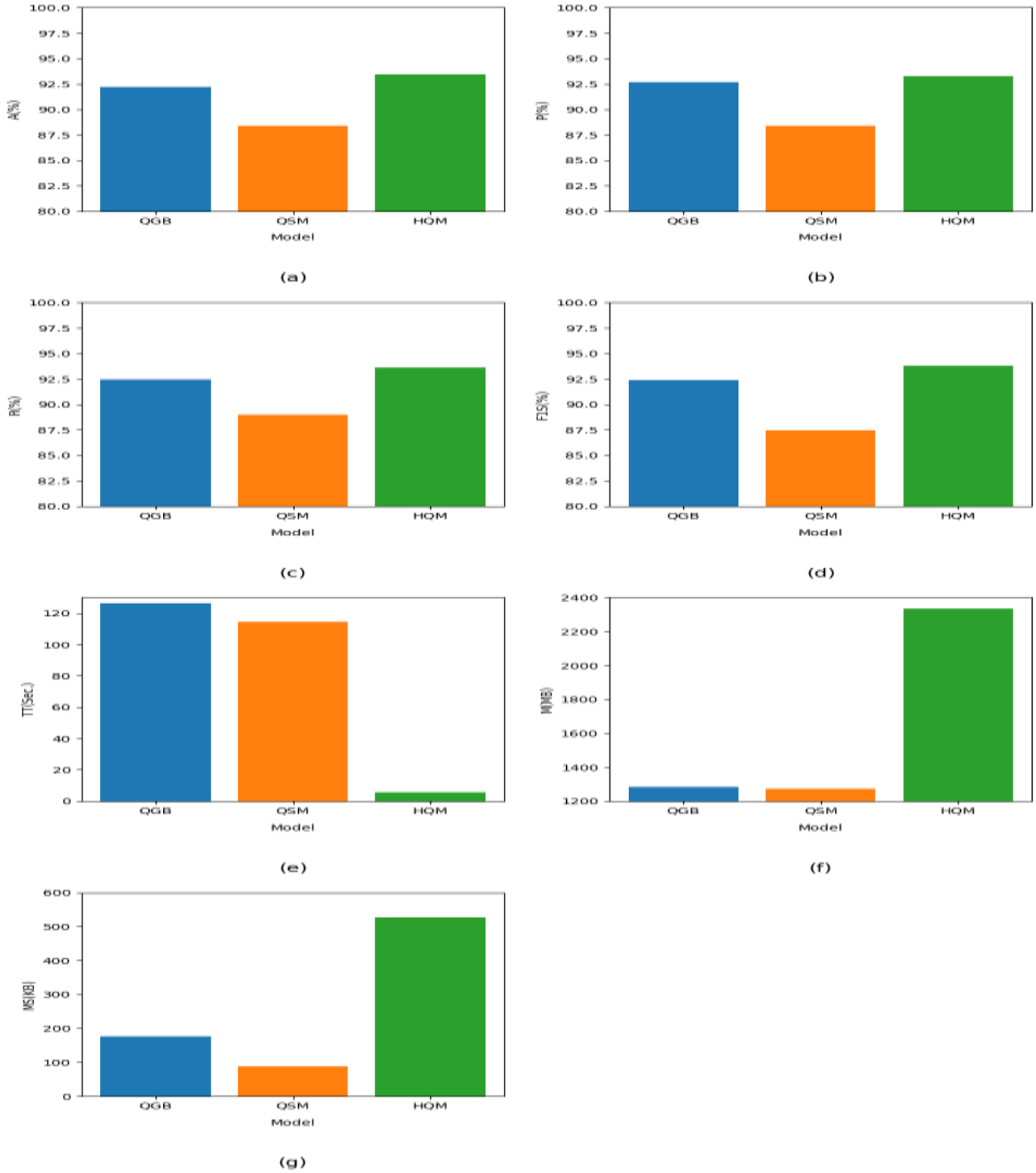


Figure 10. Results representation at 60/40 TTR before BO

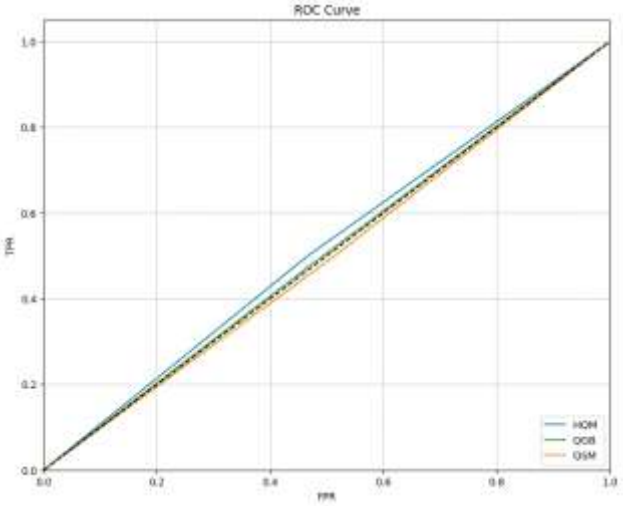


Figure 11. ROC: 60/40 TTR before BO

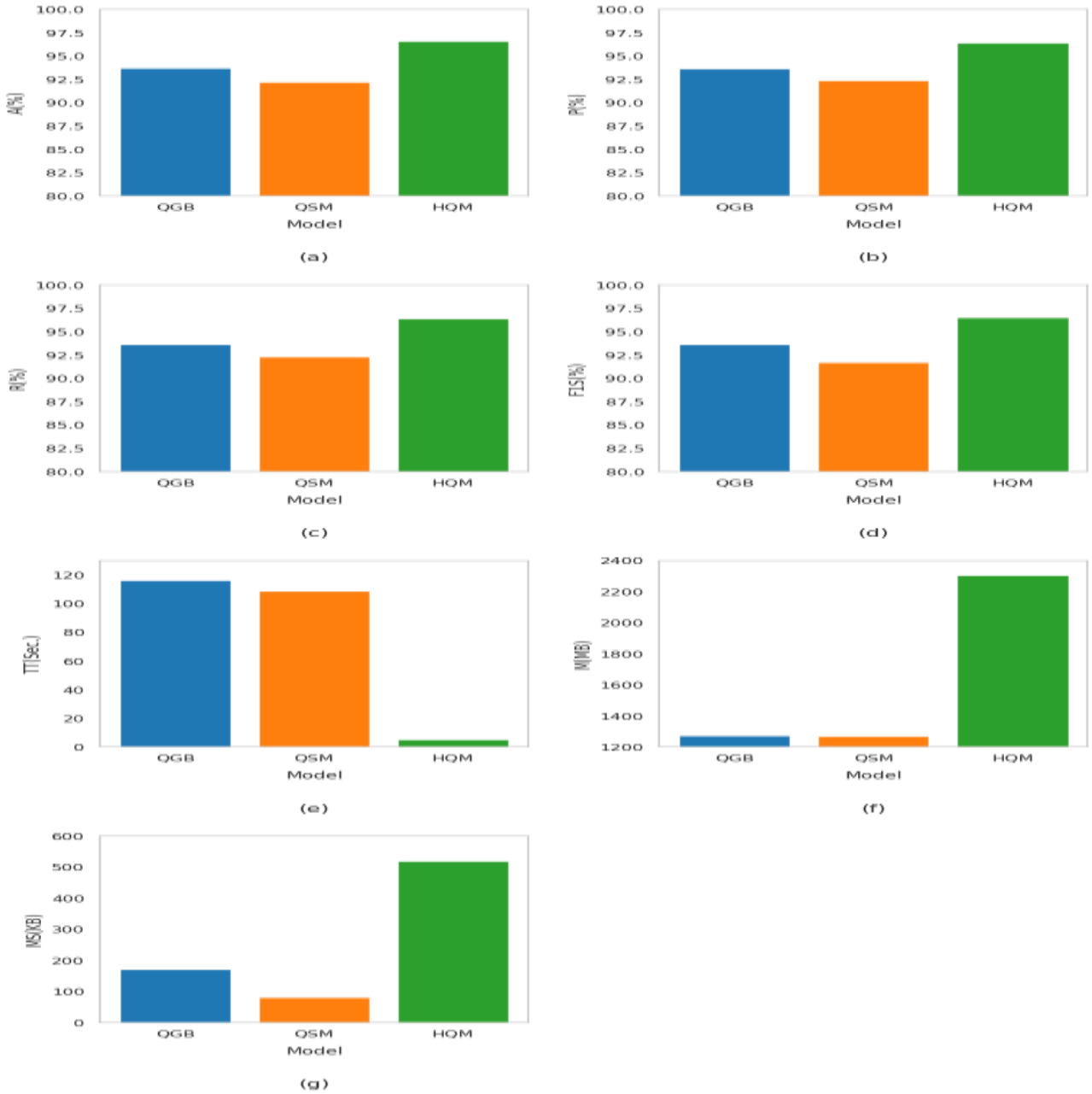


Figure 12. Results representation at 60/40 TTR after BO

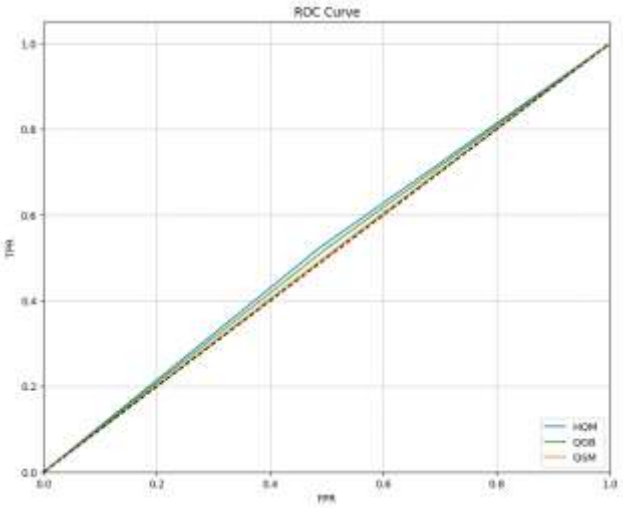


Figure 13. Results representation at 60/40 TTR after BO

Table 7: Average Performance Parameters Results Before BO for several Models

Method	A	P	R	F1S	TT	M	MS
QGB	93.48	93.64	93.29	93.52	127.32	1283.24	178.38
QSM	89.89	89.29	90.06	89.22	115.17	1275.19	89.31
HQM	95.53	95.11	95.25	95.29	5.60	2334.38	527.62

Table 8: Average Performance Parameters Results After BO for several Models

Method	A	P	R	F1S	TT	M	MS
QGB	96.55	95.60	96.16	96.49	118.06	1274.28	172.15
QSM	94.45	94.27	94.18	93.30	109.95	1270.47	81.31
HQM	97.63	97.52	97.58	97.55	5.46	2309.20	519.23

Table 7 gives the average performance parameters for these models before BO. HQM shows an average A of 95.53%, P of 95.11%, R of 95.25%, F1S of 95.29%, and TT of 5.60 sec., indicating better performance in correctly classifying positive instances. However, it requires a higher average M of 2334.38 MB and MS of 527.62 KB, which is higher than both QGB and QSM. QGB achieves moderate performance with an A of 93.48%, P of 93.64%, R of 93.29%, and F1S of 93.52%. It also has a moderate TT of 127.32 sec. and requires M of 1283.24 MB and MS of 178.38 KB. QSM, on the other hand, provides relatively lower results with an average A of 89.89%, P of 89.29%, R of 90.06%, and F1S of 89.22%. However, it has a lower TT of 115.17 sec., making it faster than QGB but slower than HQM. It uses 1275.19 MB of M and the lowest MS of 89.31 KB compared to QGB and HQM. From the above analysis, it is observed that different models provide outputs differently depending on the performance parameters. While HQM shows better performance using the metrics such as A, P, R, F1S, and TT, QSM provides better performance in terms of M and MS. Similarly, Table 8 gives the average performance parameters for these models after BO by considering these parameters.

Table 9: Incremental Results Representation by Applying BO

Method	A	P (%)	R (%)	F1S (%)	TT (Sec.)	M (MB)	MS (KB)
QGB	+3.07	+1.96	+2.87	+2.97	-9.26	-8.96	-6.23
QSM	+4.56	+4.98	+4.12	+4.08	-5.22	-4.72	-8.00
HQM	+2.10	+2.41	+2.33	+2.26	-0.14	-25.18	-8.39

Table 9 shows the incremental results after applying BO for the models QGB, QSM, and HQM. All models exhibit significant improvements across all performance metrics. HQM shows improvements with +2.10% in A, +2.41% in P, +2.33% in R, and +2.26% in F1S. Its TT decreased by 0.14 sec., M by 25.18 MB, and MS by 8.39 KB. QGB shows an increase of +3.07% in A, +1.96% in P, +2.87% in R, and +2.97% in F1S. However, it shows a decrease in TT by 9.26 sec., M by 8.96 MB, and MS by 6.23 KB. QSM shows improvements of +4.56% in A, +4.98% in P, +4.12% in R, and +4.08% in F1S. It also achieves reductions in TT by 5.22 sec., M by 4.72 MB, and MS by 8.00 KB.

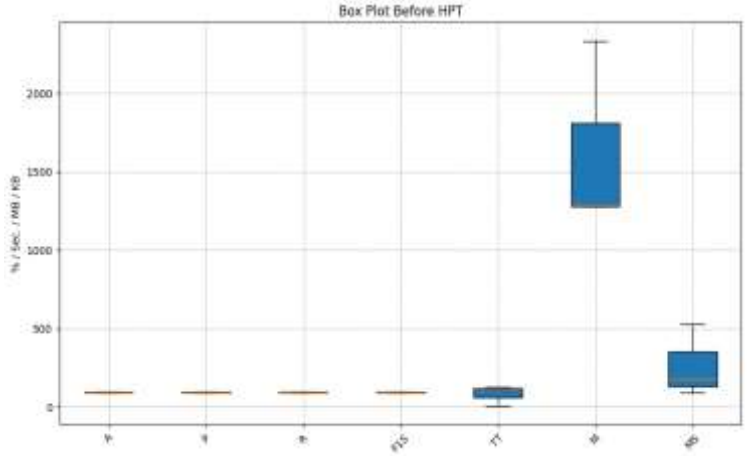


Figure 14. Boxplot before BO

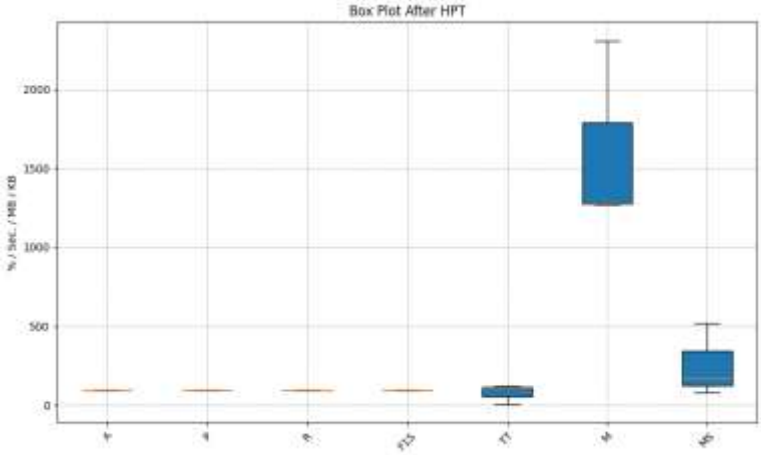


Figure 15. Boxplot after BO

In Figure 14, the boxplot illustrates the distribution of the average parameters before BO for the models QGB, QSM, and HQM. HQM has higher values in the range of 95.11%–95.53% for the parameters A, P, R, F1S, and 5.60 sec. for TT, so its box plot represents a narrow box with little to no variation. QGB has a comparatively lower range for these parameters with values around 93.29%–93.64%, 127.32 sec., 1283.24 MB, and 178.38 KB, respectively, so its box will reflect a wider range compared to HQM. QSM has the lowest values for A, P, R, and F1S in the range of 89.22%–90.06%, TT of 115.17 sec., and better M and MS with values of 1275.19 MB and 89.31 KB, respectively. So, its box plot shows both wider and narrower boxes depending on the metric values. Similarly, in Figure 15 the boxplot illustrates the distribution of the average parameters after BO.

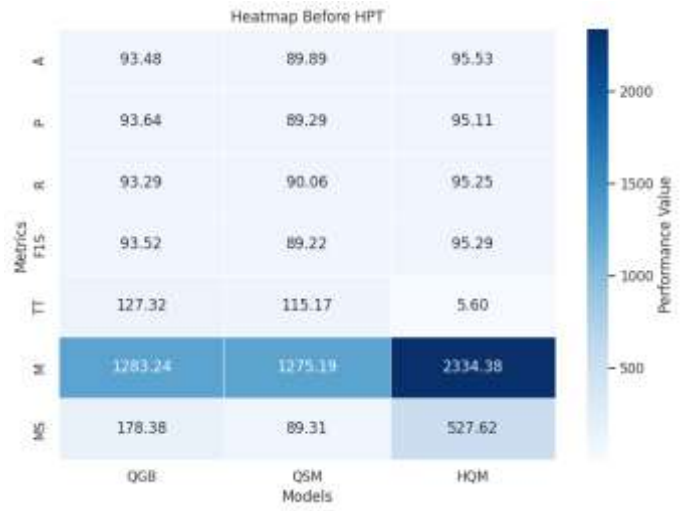


Figure 16. Heatmap before BO

Figure 17. Heatmap after BO

A heat map is a graphical representation of data where individual values are represented by colors. It is used to display the relative magnitude of metrics across models. In Figure 16, the heat map shows the performance of each model across the different parameters before BO. The heat map is color-coded to illustrate the values of each parameter. These parameters such as A, P, R, F1S, and TT have better values for HQM, exceeding 95% and 5.60 sec. accordingly, which is shown in darker or more intense colors indicating strong performance. QGB shows comparatively lighter colors, representing its slightly lower values of around 93.29%–93.64%. QGB has a comparatively lighter color indicating a higher TT, moderate M, and MS. QSM falls somewhere in between but it shows a darker color representing better M and MS with values 1275.19 MB and 89.31 KB accordingly, whereas HQM and QGB have lighter colors for M and MS. In Figure 17, the heat map shows the performance of each model across the different parameters after BO.

6.STATISTICAL ANALYSIS

To statistically evaluate the effect of BO on QGB, QSM, and HQM, a comparative analysis of mean performance differences is conducted across all evaluation metrics: A, P, R, F1S, TT, M, and MS (Table 10). Since each model is evaluated under two conditions (before and after BO), a repeated-measures comparative structure is implicitly formed, allowing analysis of variance in performance trends across optimization states.

6. 1. Mean Performance Variation Analysis

Table 11 presents the mean performance change after BO for QGB, QSM, and HQM across all evaluation metrics. It is observed that all models exhibit consistent improvement in classification performance, as indicated by positive gains in A, P, R, and F1S. Among them, QSM shows the highest relative improvement with $\Delta A = +4.56\%$, $\Delta P = +4.98\%$, $\Delta R = +4.12\%$, and $\Delta F1S = +4.08\%$, indicating that BO has the strongest impact on the comparatively weaker baseline model. QGB also demonstrates steady improvement across all metrics, while HQM shows moderate but stable gains, reflecting its already high baseline performance. In terms of computational efficiency, all models exhibit negative changes in TT, M, and MS, indicating reductions after BO. HQM achieves the largest reduction in memory usage ($\Delta M = -25.18$ MB), while QGB and QSM also show noticeable reductions in TT and MS. Overall, the results confirm that BO enhances predictive performance while simultaneously improving computational efficiency across all models.

Table 10 Mean Performance Change After BO (Δ = After − Before)

Method	ΔA	ΔP	ΔR	ΔF1S	ΔTT	ΔM	ΔMS
QGB	+3.07	+1.96	+2.87	+2.97	-9.26	-8.96	-6.23
QSM	+4.56	+4.98	+4.12	+4.08	-5.22	-4.72	-7.99
HQM	+2.10	+2.41	+2.33	+2.26	-0.14	-25.18	-8.39

7.CONCLUSION

This work focuses on the use of QML to predict the LCS (MI, MD, SV). The models used are QGB, QSM, and HQM to track the LCS. The models are compared using the following metrics: A, P, R, F1S, TT, M and MS based on the TTR distribution of 80:20, 70:30, and 60:40 both before and after BO. BO is applied to improve model performance. Using ROC, boxplot and heatmap graphs, these metrics will be visualized to highlight the results observed. The results show that HQM has high A, P, R, F1S and TT scores with average values of 95.53, 95.11, 95.25, 95.29, and 5.60 before BO. After BO, HQM has scores of 97.63, 97.52, 97.58, 97.55 and 5.46 respectively. However, QSM performs better in terms of M and MS with values 1275.19 and 89.31, and 1270.47 and 81.31 accordingly as compared to QGB and HQM. This

work can also be extended to apply FL-based models on several datasets to analyze the performance. This work can also be extended to apply edge ML-based models on several datasets to analyze the performance.

REFERENCES

1. Li T, Li J, Jiang H, Skiles DB. Deep learning prediction of drug-induced liver toxicity by manifold embedding of quantum information of drug molecules. Pharm Res. 2025;42(1):109-122.
2. Ayalew AM, Degife WA, Asnake NW, Nibret EA, Bezabh YA, Abuhayi BM, et al. Improved sarcoidosis disease detection using deep learning and histogram of oriented gradients with quantum SVM. Discov Appl Sci. 2025;7(3):154.
3. Thoufeeq JM, Prakasarao A, Ganesan S, Rajasekar JS, Rammohan A, Rela M. Excited state kinetics of tryptophan and NAD(P)H in blood plasma of normal and abnormal liver conditions: A tool to understand the metabolic changes and classification. Spectrochim Acta A Mol Biomol Spectrosc. 2025;125867.
4. Lusnig L, Sagingalieva A, Surmach M, Protasevich T, Michiu O, McLoughlin J, et al. Hybrid quantum image classification and federated learning for hepatic steatosis diagnosis. Diagnostics. 2024;14(5):558.
5. Pomarico D, Monaco A, Amoroso N, Bellantuono L, Lacalamita A, La Rocca M, et al. Emerging generalization advantage of quantum-inspired machine learning in the diagnosis of hepatocellular carcinoma. Discov Appl Sci. 2025;7(3):1-19.
6. Wang K, Margolis S, Cho JM, Wang S, Arianpour B, Jabalera A, et al. Non-invasive detection of early-stage fatty liver disease via an on-skin impedance sensor and attention-based deep learning. Adv Sci. 2024;11(31):2400596.
7. Rahi P, Kang SS, Singh AP, Singh I. Liver disease risk prediction using regularized deep learning: A novel approach. In: Proceedings of the 2024 2nd International Conference on Advancements and Key Challenges in Green Energy and Computing (AKGEC); 2024. p. 1-7.
8. Bandaru SC, Mohan GB, Kumar RP, Altalbe A. Swingale: fusion of Swin transformer and attention mechanism for GAN-augmented liver tumor classification with enhanced deep learning. Int J Inf Technol. 2024;16(8):5351-5369.
9. Wu Q, Yu J, Zhang M, Xiong Y, Zhu L, Wei B, et al. Serum lipidomic profiling for liver cancer screening using surface-assisted laser desorption ionization MS and machine learning. Talanta. 2024;268:125371.
10. Potapova EV, Shupletsov VV, Dremin VV, Zherebtsov EA, Mamoshin AV, Dunaev AV. In vivo time-resolved fluorescence detection of liver cancer supported by machine learning. Lasers Surg Med. 2024;56(10):836-844.
11. Vijayakumar S. A hybrid QNN-based framework for accurate early detection of HCV liver abnormalities from CT scans using custom transfer learning and AI edge device. In: Proceedings of the 2023 IEEE Region 10 Symposium (TENSYMP); 2023. p. 1-5.
12. Sawant LD, Ritti R, Harshith N, Kodipalli A, Rao T, Rohini BR. Analysis and prediction of liver cirrhosis using machine learning algorithms. In: Proceedings of the 2023 3rd International Conference on Intelligent Technologies (CONIT); 2023. p. 1-5.
13. Prakash NN, Rajesh V, Namakhwa DL, Pande SD, Ahammad SH. A DenseNet CNN-based liver lesion prediction and classification for future medical diagnosis. Sci Afr. 2023;20:e01629.
14. Suárez M, Martínez R, Torres AM, Torres B, Mateo J. A machine learning method to identify the risk factors for liver fibrosis progression in nonalcoholic steatohepatitis. Dig Dis Sci. 2023;68(9):3801-3809.
15. Guarneros-Nolasco LR, Alor-Hernández G, Prieto-Avalos G, Sánchez-Cervantes JL. Early identification of risk factors in non-alcoholic fatty liver disease (NAFLD) using machine learning. Mathematics. 2023;11(13):3026.
16. Zhang L, Xiao Z, Jiang W, Luo C, Ye M, Yue G, et al. Liver fibrosis MR images classification based on higher-order interaction and sample distribution rebalancing. Health Inf Sci Syst. 2023;11(1):51.
17. Reddy KP, Satish M, Prakash A, Babu SM, Kumar PP, Devi BS. Machine learning revolution in early disease detection for healthcare: advancements, challenges, and future prospects. In: Proceedings of the 2023 IEEE 5th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA); 2023. p. 638-643.
18. Pahari S, Choubey DK. Analysis of liver disorder by machine learning techniques. In: Soft Computing: Theories and Applications. Proceedings of SoCTA 2020. Singapore: Springer; 2021. p. 587-601.
19. Ullah U, Garcia-Zapirain B. Quantum machine learning revolution in healthcare: a systematic review of emerging perspectives and applications. IEEE Access. 2024;12:11423-11450.
20. Lakshmipriya B, Pottakkat B, Ramkumar G. Deep learning techniques in liver tumour diagnosis using CT and MR imaging: a systematic review. Artif Intell Med. 2023;141:102557.
21. Maheshwari D, Garcia-Zapirain B, Sierra-Sosa D. Quantum machine learning applications in the biomedical domain: a systematic review. IEEE Access. 2022;10:80463-80484.
22. Grignaffini F, Barbuto F, Troiano M, Piazzo L, Simeoni P, Mangini F, et al. The use of artificial intelligence in the liver histopathology field: a systematic review. Diagnostics. 2024;14(4):388.
23. Vasanthakumar GU, Singh M. Quantum machine learning in healthcare: diagnostics and drug discovery. In: Quantum Machine Learning: Quantum Algorithms and Neural Networks. 2024. p. 39.
24. Kumar P, Dwivedi P. Diagnosis of liver disorder. In: Computational Intelligence for Genomics Data. Amsterdam: Elsevier; 2025. p. 199-224.

25. Nivetha G, et al. Quantum machine learning for the future medical analysis: A comprehensive review and future prospects. In: Proceedings of the 2024 9th International Conference on Communication and Electronics Systems (ICCES); 2024. p. 1256-1261.
26. Hu N, Yan G, Tang M, Wu Y, Song F, Xia X, Chan LWC, et al. CT-based methods for assessment of metabolic dysfunction associated with fatty liver disease. Eur Radiol Exp. 2023;7(1):72.
27. Roy AK, Jena KK, Mohapatra D. Quantum inspired data driven fitness monitoring approach for industrial machinery. J Inst Eng India Ser B. 2026:1-18.
28. Kaggle. Liver cirrhosis stage classification dataset. Available from: https://www.kaggle.com/datasets/aadarshvelu/liver-cirrhosis-stage-classification. [Accessed on Dec. 2025]