Quantum Machine Learning

Investigating kernel-based Quantum Random Forests
for improved classification

Henricus Bracht

September 2023

1 Introduction

With the advances that could be achieved in the past years in the realm of Quantum Computing, it becomes ever more important to develop algorithms that can harness this Quantum Advantage over classical computing. Intuitively, the first step to approach this is to go back to the realm of classical computing and transfer paradigms in Classical Machine Learning to Quantum Computing. One such paradigms are Random Forests, that are unique by being constituted by many weak independent classifiers that emerge in combining their output to an expressive model. Classical Random Forests represent a robust model with numerous advantages. This implies the necessity of an analogous algorithm that can be employed in the Quantum realm. The paper at hand deals with this novel approach that they call Quantum Random Forests. In the following assignment research, I will summarize this paper, delivering main hypothesis, methodologies, findings. Departing from that, I will elaborate on potential strengths and weaknesses of the paper and finally I will propose possible improvements or extensions to the paper. I implemented one of which, that can be found in this GitHub repository.

2 Summary

2.1 Main Objectives and Motivations of the Research

2.1.1 Objectives

This research project is centered around advancing Quantum Machine Learning methods, with a primary focus on overcoming limitations found in existing approaches, particularly Quantum Support Vector Machines.

A significant part of this endeavor involves the creation of a Quantum Random Forest, a novel quantum model that distinguishes itself from established quantum models like Quantum Support Vector Machines and Quantum Neural

Networks. The Quantum Random Forest is designed to utilize the strengths of Quantum Decision Trees exploiting ensemble methods that make use of bagging.

To ensure the reliability and robustness of the proposed Quantum Random Forests, the research provides essential generalization error bounds for the model. These bounds offer theoretical assurances that help restrict errors coming from finite sampling on the Nyström Quantum Kernel Estimation strategy.

Efficiency improvement is another core objective, with a focus on reducing sampling complexity compared to Quantum Kernel Estimation. The development and assessment of the Nyström Quantum Kernel Estimation strategy plays a decisive role in achieving this goal.

The research also aims to demonstrate that the Quantum Random Forest can effectively address multi-class supervised problems without needing additional computations, a notable advantage over Quantum Support Vector Machines.

Furthermore, the research introduces the capacity of the Quantum Random Forests to produce probabilistic outputs, enabling class label likelihood estimation. This feature enhances the utility of the Quantum Random Forests in a wide range of machine learning tasks.

Lastly, the study elaborates on the relationships between model hyperparameters for the quantum split function. Through this exploration, the research seeks to optimize the Quantum Random Forest for specific datasets and types of problems, further enhancing its effectiveness in real-world applications.

2.1.2 Motivations

The driving force behind this research is the need to overcome the limitations connected to current quantum machine learning models, particularly Quantum Support Vector Machines (QSVMs). These limitations, ranging from vanishing kernel elements to scalability issues, underscore the necessity of seeking a more practical and efficient quantum learning approach.

In pursuit of this goal, the research is motivated by wanting to develop a quantum model with enhanced expressiveness. Drawing inspiration from classical machine learning techniques like random forests, the aim is to create quantum model hypotheses that offers distinct advantages over linear quantum models.

Given that the field of quantum machine learning is still in its early stages, there is a strong motivation to make quantum learning methods more practical and applicable in real-world scenarios. This involves tackling challenges related to quantum state preparation and data access, ensuring that these methods can be effectively employed beyond theoretical settings.

Another motivating factor is the quest to improve both generalization performance and computational efficiency in quantum models. Strategies such as Nyström Quantum Kernel Estimation and the Nyström approximation play a crucial role in achieving this objective.

Furthermore, the research seeks to address the demand for quantum models capable of seamlessly handling multi-class problems and providing probabilistic outputs. These capabilities are highly valuable in diverse machine learning applications, further highlighting the relevance and significance of this research endeavor.

2.2 Key methodologies and techniques used

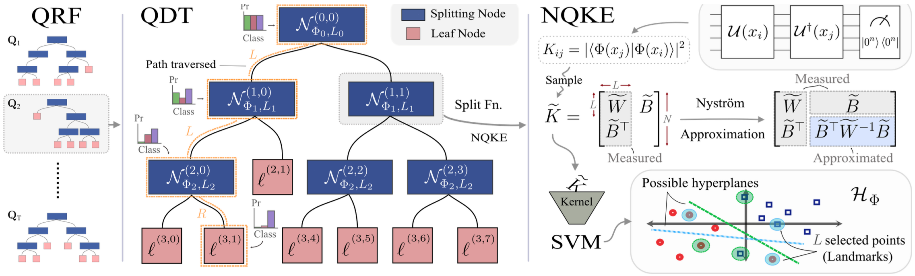
QRF
Q_1
Q_2
Q_T
QDT
Pr
Class
$\mathcal{N}^{(0,0)}_{\Phi_0,L_0}$
Splitting Node
Leaf Node
Path traversed
L
$\mathcal{N}^{(1,0)}_{\Phi_1,L_1}$
$\mathcal{N}^{(1,1)}_{\Phi_1,L_1}$
Split Fn.
NQKE
$\mathcal{N}^{(2,0)}_{\Phi_2,L_2}$
$\ell^{(2,1)}$
$\mathcal{N}^{(2,2)}_{\Phi_2,L_2}$
$\mathcal{N}^{(2,3)}_{\Phi_2,L_2}$
R
$\ell^{(3,0)}$
$\ell^{(3,1)}$
$\ell^{(3,4)}$
$\ell^{(3,5)}$
$\ell^{(3,6)}$
$\ell^{(3,7)}$
NQKE
$K_{ij} = |\langle\Phi(x_j)|\Phi(x_i)\rangle|^2$
$\mathcal{U}(x_i)$
$\mathcal{U}^\dagger(x_j)$
$|0^n\rangle\langle 0^n|$
Sample
Measured
$\widetilde{K} = \begin{bmatrix} \widetilde{W} & \widetilde{B} \\ \widetilde{B}^\top & \end{bmatrix}$
L
N
Nyström
Approximation
$\begin{bmatrix} \widetilde{W} & \widetilde{B} \\ \widetilde{B}^\top & \widetilde{B}^\top\widetilde{W}^{-1}\widetilde{B} \end{bmatrix}$
Measured
Approximated
$\widetilde{K}$
Kernel
SVM
Possible hyperplanes
$\mathcal{H}_\Phi$
L selected points
(Landmarks)

Figure 1: Depiction of the mode of operation of Quantum Random Forests

2.2.1 Quantum Random Forest

A Quantum Random Forest has similarities with a Classical Random Forest but operates differently due to its different constituent Quantum Decision Trees.

In essence, a Quantum Random Forest, much like a conventional Random Forest, consists of T weak independent classifiers. However, what sets it apart is the type of classifiers it employs. Instead of using traditional Decision Trees like Random Forests, Quantum Random Forests rely on Quantum Decision Trees.

Quantum Decision Trees are trained using only a subset or partition of the dataset. This procedure not only serves as a form of regularization but also significantly reduces the time complexity of the algorithm. This specialized training technique is commonly referred to as ”bagging” and is commonly used for ensemble methods.

When it comes to making predictions with a Quantum Random Forest, the process is distinct. The final prediction is obtained by averaging the outcomes generated by all the Quantum Decision Trees in the forest, each of which provides a probability distribution, whereas Classical Random Forests with Classical Decision Trees yield a distinct prediction result.

To evaluate the performance of a Quantum Random Forest, it’s typically tested using a separate test dataset. This testing phase allows us to assess the accuracy and effectiveness of the Quantum Random Forest in making predictions on new, unseen data.

2.2.2 Quantum Decision Trees

Our Quantum Random Forests are constructed using Quantum Decision Trees, which function as weak independent classifiers following a structure similiar to binary decision trees. These Quantum Decision Trees have nodes that can either further split the tree or serve as leaf nodes, determining the final output of the tree.

The effectiveness of a split within these trees is assessed using a metric called Information Gain, which quantifies the reduction in entropy or disorder within a specific dataset as a result of the split. This process continues until the Quantum Decision Tree reaches its leaf nodes, each of which provides a probability distribution based on the remaining data points belonging to different classes.

A leaf node is reached when one of the following conditions is met:

- All training data points are classified as a single class.
- The number of remaining data points fall below a user-defined threshold.
- The tree reaches a depth specified by the user.

When classifying new data, we traverse down the tree, following the path until we reach a leaf node, which then produces a probability distribution for the classification.

The key to achieving optimal performance lies in ensuring that the nodes split effectively. To accomplish this, it's crucial to define an appropriate split function guided by the right hyperparameters. The goal is to maximize Information Gain, as an effective split function will improve the separation between different classes.

In practice, it's practical to use a relatively simple split function to avoid overfitting. To perform these splits, we employ a support vector machine. However, due to potential non-linearity in the data, we introduce a kernel to enable a linear hyperplane to divide the data accordingly. It's important to note that employing a highly complex method could lead to overfitting, so we introduce some level of noise by approximating the kernel.

The crux now lies in the fact that we use a Quanutm Support Vector Machine. This Quantum Support Vector Machine differs insofar from Classical Support Vector Machines that they employ a Quantum Kernel. This Quantum Kernel is determined by a Nyström Quantum Kernel Estimation.

2.2.3 Nyström Quantum Kernel Estimation

The Kernel used in our Support Vector Machine is a matrix calculated through a combination of Quantum Kernel Estimation and Nyström approximation. This approach not only saves computational time but also ensures generalization.

First, we select a subset of Landmark points, denoted as S_L, from the original dataset at the split, $S_L \subseteq S$, where $L < N$. We then apply the Quantum Kernel Estimator by computing inner products between feature vectors of elements in S and S_L, as defined by

$$k_\phi(x', x'') = |\langle\Phi(x'')|\,|\Phi(x')\rangle|^2 = |\langle 0|\, U(x'')U^\dagger(x')\,|0\rangle|^2$$

This process results in an incomplete Gram matrix, represented as $G_{ij} = k(x_i, x_y) \coloneqq \begin{pmatrix} W & B \end{pmatrix}^T$. To complete this matrix, we employ the Nyström Approximation. Since the Gram matrix is symmetrical, it ultimately yields a complete Kernel Matrix.

$$\hat{K} = \begin{pmatrix} W & B \\ B^T & B^T W^{-1} B \end{pmatrix}$$

In simpler terms, the Nyström method relies on relationships among selected columns of the kernel matrix to create a simplified version with fewer columns. While this simplified version may not perform well when the original matrix is nearly full rank, the Nyström method operates on the assumption that data often follows patterns and doesn't utilize all available options. Thus, a simplified matrix with fewer columns can still yield reasonable results.

Importantly, the Kernel Matrix remains positive semi-definite, preserving the convex nature of the Support Vector Machine problem.

The split function employed can be described as

$$\mathcal{N}_{\Phi;\alpha} = \mathrm{sign}(\sum_{i=1}^{N} \alpha_i \tilde{y}_i \hat{k}_\Phi(x, x_i))$$

using $\tilde{y}_i = F(y_i)$, with F representing either a "one against all" or "even split" strategy.

The creation of such split functions is designed to minimize correlation between Quantum Decision Trees because they are constructed based on landmark data points. Additionally, we have the flexibility to adjust hyperparameters L and Φ.

The approach of utilizing different kernels at each depth of the tree enhances the Quantum Support Vector Machines beyond a simple ensemble. However, it's worth noting that this approach may require increased computational efforts due to its complexity.

2.3 Results and Findings

2.3.1 Theoretical results

To develop effective strategies for constructing Quantum Learning Algorithms, it's crucial to assess the bounds of generalization errors. The primary challenge arises from training only on set of samples from the underlying distribution, which allows us to compute the error for these known points, referred to as the training error. However, this training error alone doesn't guarantee the model's capability to handle new, unseen data points effectively.

Therefore, it becomes imperative to assess the error associated with unseen data points, known as the generalization error. Given a certain amount of correctly classified data points, with almost complete certainty, we can bound the error. This lemma underscores the importance of Quantum Decision Trees with split nodes that create clear distinctions i.e., larger margins, and have lower model complexity. These Quantum Decision Trees are more likely to generalize well, making them correctly handle new, unseen data.

Furthermore, the Kernel matrix may suffer from sampling errors, as indicated in Lemma 2. While increasing the number of Landmark points can improve sampling accuracy, it can also compromise the accuracy of the Nyström algorithm, as demonstrated in the second term. It's important to note that the introduction of error through the Nyström Approximation is deliberate and serves to weaken the constituting classifiers intentionally.

Lemma 3 suggests that the error coming from sampling itself can be bounded. However, it's essential to understand that having a function or kernel that is closer to the optimal choice doesn't guarantee improved model performance.

In the context of Quantum Random Forests, they exhibit the ability to learn from data that classical learning methods find challenging due to the complexity of the problem (NP Hard Problems). This capacity extends to linear quantum learners as well, like Quantum Support Vector Machines or Linear Quantum Neural Networks, demonstrating that they are capable of dealing with non-linear data.

2.3.2 Numerical results

Effective hyperparameter tuning is a crucial aspect of model optimization, but it can be challenging due to potential codependencies between these parameters.

The preprocessing steps involve applying PCA, normalization, and subsequent relabeling to the data.

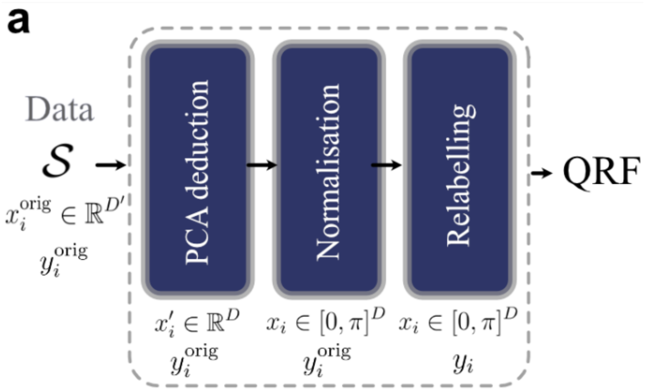

Figure 2: Preprocessing

Increasing the number of landmark points tends to weakly correlate with improved performance, and similarly, a higher number of shots leads to better results.

Interestingly, it appears that having a smaller number of landmark points may make a higher maximum tree depth more feasible.

A larger number of trees in the ensemble typically results in better overall performance.

When comparing the performance of One Against All and Even Split strategies, it becomes evident that Even Split consistently outperforms the former.

In this example of Quantum Random Forests, we can observe that ensemble machine learning techniques achieve their highest effectiveness when the weak classifiers within them exhibit some level of correlation with the expected class labels while simultaneously avoiding high correlation with each other.

In contrast to other learning models, that Quantum Random Forests were compared to i. e., Classical Support Vector Machines, Classical Random Forests, Quantum Support Vector Machines, Quantum Random Forests show their superiority in scenarios involving relabeling. However, when keeping the original labels, classical models tend to outperform Quantum Random Forests.

2.3.3 Implications

This paper essentially serves as an extension of the established Quantum Support Vector Machines, which are characterized by their use of kernels generated through Quantum Kernel Estimation.

While this extension does not introduce entirely new paradigms within Quantum Computing and Quantum Machine Learning, it represents a link that connects Quantum Support Vector Machines to methods of Classical Machine Learning, like Decision Trees and finally Random Forests.

However, it has proven difficult to create a different, completely analogous Random Forest implementation within the realm of Quantum computing. These challenges arise from fundamental limitations, such as the no-cloning theorem, which impose constraints on the feasibility of certain quantum algorithms.

3 Analysis

3.1 Strengths

The novel approach we're introducing is the integration of decision trees as weak classifiers within a larger framework known as a Quantum Random Forest. This departure from conventional methods brings several significant advantages to the field of quantum machine learning.

One key differentiator is the introduction of non-linearity into the model. Unlike traditional quantum machine learning techniques such as Quantum Neural Networks and Quantum Support Vector Machines, the Quantum Random Forest embraces non-linearity. This enhancement empowers one to tackle complex problems that would be beyond the reach of linear quantum models.

With Quantum Decision Trees, one is capable of implementing multiple hyperplanes to address problems that would be considered unlearnable with a single linear Quantum model. This expansion of capabilities is particularly evident in challenging scenarios like the Torus example i.e, seperating a 2D log space into 4 regions.

Moreover, the paper's approach incorporates a probabilistic implementation, which greatly facilitates multi-class classification problems. This feature opens up a wider range of applications and makes the model more versatile compared to traditional linear models.

A notable innovation is the estimation of the kernel within our model. This decision was made intentionally to keep the individual classifiers weak, preventing them from becoming overly expressive and prone to overfitting. To achieve this, the Nyström Kernel Estimation technique is employed, which not only contributes to better overall performance but also ensures that the model's complexity doesn't grow quadratically with the number of training instances. In essence, this approach allows one to achieve two important goals simultaneously: improved computational efficiency and reduced generalization error.

Critically, the research is supported by the ability to guarantee generalization error bounds. This means that one can confidently assert that the model doesn't introduce new errors during its operation, adding a layer of reliability to its utility.

To adapt Quantum Random Forests to diverse requirements, various hyperparameters offer flexibility in their configuration, enabling fine-tuning to suit specific needs. This flexibility ensures that Quantum Random Forests can be effectively applied across a wide spectrum of real-world problems.

3.2 Weaknesses

The paper's evaluation of the model's validity involves comparing it to other established models as I mentioned before, including Quantum Support Vector Machines, Classical Random Forests, and Support Vector Machines. The results of this comparison reveal a notable trend: In general, when applied to a classical dataset without relabeling, classical methods tend to outperform Quantum Random Forests. The only exception is observed with the Hardware-Efficient Ansatz, where Quantum Random Forests occasionally demonstrate superior performance. This observation raises questions about the necessity of Quantum Random Forests in scenarios where we want to keep the original labels and have no to little noise.

Furthermore, it seems that the paper aims to propose a solution without a clearly defined use case in mind. While striving to establish an analogous approach to Random Forests in the Quantum Realm, the paper falls short of providing a compelling use case where Quantum Random Forests might offer a distinct advantage.

A methodological weakness in the paper lies in its limited scope of comparison, focusing solely on Classical Random Forests, Quantum Support Vector Machines, and Classical Support Vector Machines. These models are either equally or less sophisticated than Quantum Random Forests. A more informative comparison would involve evaluating Quantum Random Forests against more advanced models, such as Neural Networks or hybrid models combining Quantum Neural Networks (QNN) and Convolutional Neural Networks (CNN).

Another concern arises from the fact that this algorithm is primarily designed for use with classical data. Consequently, there is limited potential for improvement with more advanced Quantum Computers, potentially only resulting in marginal speedup benefits. Likewise, the possibility to leverage on Quantum Data appears constrained.

Regarding the concept of Quantum Advantage, the critical factor is the Quantum Kernel Estimation, as it is the sole component in the Quantum Random Forest algorithm that could potentially provide such an advantage. However, it remains uncertain whether Quantum Advantage can be realized, as it can only be achieved when the kernel is hard to be estimated classically, as Havlíček has shown [4]. However, this is not a sufficient condition for Quantum Advantage. Thus, it remains questionable if reserved to few problems that we can achieve Quantum Advantage [8].

Another noteworthy drawback is the complexity and challenges associated with hyperparameter tuning in Quantum Random Forests. While this complexity allows for model customization based on specific dataset requirements, it also presents a double-edged sword, potentially making the tuning process very complex and less user-friendly.

In addition, Quantum Random Forests appear to compromise some of the advantages offered by classical Random Forests, such as robustness without extensive hyperparameter tuning and interpretability.

3.3 Improvement Proposals

In essence, enhancing Nyström Quantum Kernel Estimation is a challenging task, given its ability to improve computational efficiency by only calculating a part of the kernel matrix while simultaneously enhancing accuracy by reducing the expressiveness of constituent classifiers this way. The paper suggests an improvement approach for Nyström Quantum Kernel Estimation by employing a low-rank decomposition method for the selection of landmark points, thus reducing computational and storage demands. One example of such a method is the incomplete Cholesky decomposition with column pivoting.

Additionally, the paper explores the possibility of using different kernels along the Quantum Decision Tree, offering a potential option for optimization. For better generalization and handling larger datasets, the paper suggests training each node with different subsets of the data.

The introduction of pruning methods is proposed in the paper as a means of improving performance by removing unnecessary branches of the tree. Another enhancement idea involves weighting the tree outputs based on their performance on a test dataset.

To address scalability and adaptability to new data, the Bayesian Method is considered as an option. Currently, the Quantum Random Forest is fixed once trained, requiring complete retraining for adjustments to new data. The Bayesian algorithm can mitigate this issue.[2]

Exploring alternative low-rank matrix approximations beyond Nyström, such as randomized feature maps, is another possible way. This approach could offer computational efficiency but may introduce approximation errors. Whether this benefits the model by facilitating generalization or introduces unnecessary noise can become a subject of discussion and needs to be investigated further.

Furthermore, extending Quantum Random Forests could be extended introducing Gradient Boosted Trees[10] or other ensemble methods that combine weak tree classifiers to make predictions. Gradient Boosted Trees e. g. build a sequence of decision trees, each correcting errors made by previous trees, and their predictions are combined through weighted voting, iterating until predefined stopping conditions are met.

4 Implementation

Finding a project that could run smoothly on my Apple Silicon Mac proved to be quite a challenge. Like many projects I attempted, this one initially posed compatibility issues. However, I managed to overcome these hurdles and created the Submission Notebook as the frontend to showcase the Quantum Random Forest's capabilities. The significant improvements, which I'll elaborate on later in this chapter, lie within the .py scripts.

The main compitability issue that needed I needed to cope with was related to the parallelization library, which initially caused problems. To ensure smoother execution, I decided to run the code sequentially, even though this did compromise computational speed. Nevertheless, it still performed efficiently enough for the specific classification task at hand, involving the Breast Cancer dataset from Sklearn.

To outline the process, it begins with data loading and preprocessing. The preprocessing algorithm offers optional dimensionality reduction (if specified by X_dim), followed by normalization and encoding. The data is then divided into training and testing sets, based on the designated proportion ($train\_prop$). This preprocessed data is structured into pandas DataFrames, simplifying subsequent machine learning tasks. The function returns both the training and testing datasets.

Subsequently, the Quantum Random Forest object is defined and trained using the previously mentioned methods. For more in-depth insights into this process, you can refer to the Key Methodologies and Techniques section.

The results are put to the test using the Quantum Random Forest's testing algorithm. This algorithm takes input data with instances and labels, utilizes the Quantum Random Forest model for predictions (classification or regression), and calculates the corresponding evaluation metric (accuracy or ME). Depending on the chosen options, it may also return the predictions. In our case, we focus on classification, which yields accuracy, and we retrieve the predictions.

Finally, we conclude by printing the classification report, utilizing Sklearn's functionality. Additionally, we provide visual representations, including the tree and the correlation matrix of the trees for different classes, enhancing our understanding of the model's performance.

4.1 My own improvement

I found it most practical to implement the suggested improvement involving the selection of landmark points using the incomplete Cholesky decomposition with column pivoting. This choice was driven by the fact that it was originally proposed by the authors of the study themselves, making it a more promising approach in my view.

In essence, this implementation replaces the random sampling of landmark points with a method based on Incomplete Cholesky Decomposition. This approach enhances the selection process by capturing crucial information about the dataset, ensuring that the chosen landmark points retain the most critical aspects of the data.

To achieve this, I made modifications to the fit() method within the Nystrom Class in nystrom_cholesky.py. For a comparison with the original function, one can refer to nystrom.py. Here are the changes I made and the reasons behind them:

1. I started by calculating the covariance matrix. Understanding the covariance structure of the data is crucial, as it allows us to identify a subset of points that effectively encapsulate essential dataset information, ensuring the selected landmark points are highly informative.

2. To ensure numerical stability, I introduced the addition of a constant. Through experimentation, I discovered that without this addition, the method sometimes failed to return a positive definite matrix, which is a prerequisite for a Cholesky decomposition.

3. Next, I determined the indices of the landmark points based on their contributions to the covariance structure, ensuring that the selected points effectively capture essential dataset characteristics.

The remaining aspects of the implementation remain unchanged.

4.2 Observations

Unfortunately, I didn't observe a significant improvement in the results; overall, it performs slightly better, perhaps providing unreliable a 1% increase in accuracy. However, it's important to consider that these mathematical operations could potentially increase the algorithm's computational complexity, particularly when dealing with higher-dimensional data.

While it's possible that this modification might have some impact and potentially lead to better generalization, it's not a guaranteed or entirely reliable enhancement. Ultimately, one needs to weigh the benefits of slightly improved generalization against the trade-off of slightly increased complexity. To make this option accessible, I added *cholesky* as an attribute for use in the QuantumRandomForest class.

It's worth noting that another research paper by Romary [9] has also explored this particular improvement, indicating its relevance and potential impact in the field.

5 Conclusion

Concluding this project as a Random Forest hardcore-fan due to their simplicity, it was interesting to see such a project in the Quantum Realm. What I am mostly convinced of is the utilization and the novelty of the Nyström Quantum Kernel Estimation. It is compelling how this approach kills two birds with one stone. At the same time, it barely introduces error as it was proven by the error bounds. Moreover, it is also persuasive to have a Quantum method that employs bagging. Hence, the paper is successful in addressing their own objectives and motivations bringing utility for real-world applications. However, it still remains questionable where Quantum Random Forests can offer benefits. We have seen that the practical advantage over conventional methods is low or not existent. In my opinion, it is possible that it is a hindering approach to transfer paradigms and concepts from Classical Machine Learning to Quantum Machine Learning. Classical approaches were designed in a way to suite Classical Computers. It might therefore be useful to introduce new approaches to Quantum Machine Learning, still with the underlying idea of emergence that is evident to be powerful everywhere: Make small stupid things (neurons, trees, fish, humans) smart by combining their work (neural networks, random forests, swarms, civilizations).

References

[1] Srikumar, M., Hill, C.D., & Hollenberg, L.C. (2022). A kernel-based quantum random forest for improved classification. ArXiv, abs/2210.02355.

[2] Ji N, Bao R, Mu X, Chen Z, Yang X and Wang S (2023) Cost-sensitive classification algorithm combining the Bayesian algorithm and quantum decision tree. Front. Phys. 11:1179868. doi: 10.3389/fphy.2023.1179868

[3] Robnik-Šikonja, M. (2004). Improving Random Forests. In: Boulicaut, JF., Esposito, F., Giannotti, F., Pedreschi, D. (eds) Machine Learning: ECML 2004. ECML 2004. Lecture Notes in Computer Science(), vol 3201. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-30115-8-34

[4] Havlíček, V., Córcoles, A.D., Temme, K. et al. Supervised learning with quantum-enhanced feature spaces. Nature 567, 209–212 (2019). https://doi.org/10.1038/s41586-019-0980-2

[5] Sun, S., Zhao, J., & Zhu, J. (2015). A review of Nyström methods for large-scale machine learning. Information Fusion, 26, 36–48. https://doi.org/10.1016/j.inffus.2015.03.001

[6] Shahhosseini, M., & Hu, G. (2020). Improved Weighted Random Forest for Classification Problems. ArXiv, abs/2009.00534.

[7] West, M. T., Erfani, S. M., Leckie, C., Sevior, M. E., Lloyd, & Usman, M. (2023). Benchmarking adversarially robust quantum machine learning at scale. Physical Review Research, 5(2). https://doi.org/10.1103/physrevresearch.5.023186

[8] Liu, Y., Arunachalam, S., & Temme, K. (2020). A rigorous and robust quantum speed-up in supervised machine learning. Nature Physics, 17, 1013 - 1017.

[9] Romary, T. (2013). Incomplete Cholesky decomposition for the kriging of large datasets. Spatial Statistics, 5, 85–99. doi:10.1016/j.spasta.2013.04.0

[10] Friedman, J. H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. The Annals of Statistics, 29(5), 1189–1232. https://www.jstor.org/stable/2699986