QGAPHEnsemble : Combining Hybrid QLSTM Network Ensemble via Adaptive Weighting for Short Term Weather Forecasting

Anuvab Sen[1], Aakash Mallik[2], Ananya Sutradhar[3], Sujith Sai[4], Mayukhi Paul[5] and Chhandak Mallick[6]

[1,3,5]Indian Institute of Engineering Science and Technology, Shibpur, Howrah - 711103, India

[2]National Institute of Technology Karnataka, Surathkal - 575025, India

[4]National Institute of Technology, Rourkela, Odisha - 769008, India

[6]Jadavpur University, Kolkata - 700032, India

Email: sen.anuvab@gmail.com[1], aakashmallik.7777@gmail.com[2], ananyasutradhar1403@gmail.com[3],
ssujith.sai04@gmail.com[4], mayukhipaul03@gmail.com[5] and chhandak.mallick@icloud.com[6]

Abstract—**Accurate weather forecasting holds significant importance, serving as a crucial tool for decision-making in various industrial sectors. The limitations of statistical models, assuming independence among data points, highlight the need for advanced methodologies. The correlation between meteorological variables necessitate models capable of capturing complex dependencies. This research highlights the practical efficacy of employing advanced machine learning techniques proposing GenHybQLSTM and BO-QEnsemble architecture based on adaptive weight adjustment strategy. Through comprehensive hyper-parameter optimization using hybrid quantum genetic particle swarm optimisation algorithm and Bayesian Optimization, our model demonstrates a substantial improvement in the accuracy and reliability of meteorological predictions through the assessment of performance metrics such as MSE (Mean Squared Error) and MAPE (Mean Absolute Percentage Prediction Error). The paper highlights the importance of optimized ensemble techniques to improve the performance the given weather forecasting task.**

Index Terms—**Quantum Genetic Algorithm (QGA), Particle Swarm Optimization, Quantum Long Short Term Memory (QLSTM), Deep Learning, Ensemble Forecasting, Metaheuristics**

I. INTRODUCTION

Accurate weather forecasting is crucial in various sectors such as agriculture, transport, disaster management and public safety. The precise prediction of meteorological conditions is indispensable for informed decision-making, enabling risk mitigation and resource optimization. Amid technological advancements, researchers and meteorologists are working to improve the precision of weather prediction to furnish more dependable and timely information to relevant entities across different geographical and organizational levels. Recognizing the inherent challenges, meteorological time series data often exhibit complex temporal dependencies, non-linear patterns, and irregularities that conventional models struggle to capture effectively [1] [2]. Traditional models which assume independence among data points, may overlook the intrinsic temporal dependencies present in meteorological phenomena. In the field of meteorology, time series prediction has witnessed notable progress with a diverse array of models. Early efforts

predominantly relied on statistical methods such as autoregressive integrated moving average (ARIMA) and seasonal-trend decomposition using LOESS (STL) [3] [4]. These models, while providing a foundation for time series analysis, often struggled to capture the complexity of atmospheric phenomena characterized by non-linear trends and seasonality.

In recent years, machine learning models have gained prominence, particularly the application of ensemble techniques like Random Forests and Gradient Boosting Machines [5] [6]. The ensemble approach uses the collective strength of multiple models to improve predictive accuracy and robustness. Additionally, Support Vector Machines (SVM) and K-Nearest Neighbors (KNN) have been explored for their ability to explain patterns in meteorological time series data [7]. Recurrent Neural Networks (RNNs) have emerged as a powerful class of models for capturing sequential dependencies. Long Short-Term Memory (LSTM) networks, in particular, have demonstrated success in modeling temporal relationships and adapting to the dynamic nature of atmospheric conditions [8] [9]. Furthermore, hybrid models that combine traditional statistical methods with machine learning algorithms have been proposed to use the strengths of both paradigms [10] [11] [12]. While these models represent significant strides in meteorological time series forecasting, challenges persist in accounting for the spatial and temporal heterogeneity of atmospheric processes. Addressing these challenges requires innovative approaches that uses the potential of emerging technologies.

This paper proposes two novel optimised quantum hybrid ensemble networks BO-QEnsemble and GenHybQLSTM for the given weather forecasting task. Through extensive experimental studies, we empirically demonstrate the superiority of our proposed models over state-of-the-art (SOTA) counterparts, showcasing enhanced performance across various evaluation metrics, including Mean Absolute Percentage Error (MAPE) and Mean Squared Error (MSE). The presented findings contribute to the field of weather forecasting and establish the efficacy of our novel ensemble networks in

surpassing existing benchmarks.

II. PRELIMINARIES

This sections provides a concise overview of the fundamental concepts of Quantum Long Short-Term Memory (QLSTM), Particle Swarm Optimisation (PSO) and Bayesian Optimisation (BO) [13] [14] [15] [16]. These foundational elements are pivotal components deployed within the proposed ensemble approach.

A. Quantum Long Short-Term Memory (QLSTM)

QLSTM is the quantum extension of LSTM, a variety of the recurrent neural network (RNN) architecture which serves as one of the most efficient models for handling sequential data, making them the perfect candidate for time-series forecasting tasks [17]. Figure 1. illustrates the QLSTM architecture.
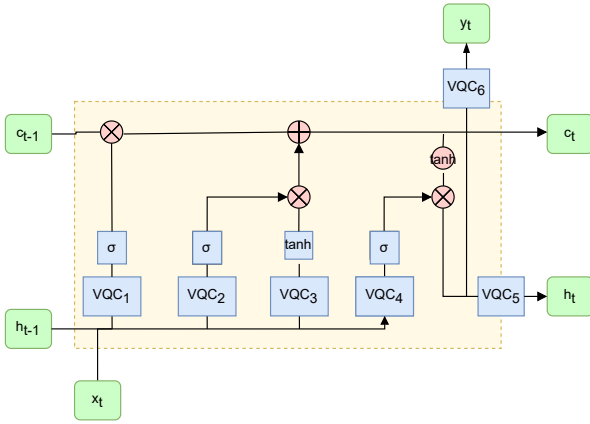

Fig. 1: QLSTM architecture (Variational Quantum Circuits)

The mathematical formulation of a QLSTM cell is given as follows:

$$f_t = \sigma(VQC_1(v_t)$$

$$i_t = \sigma(VQC_2(v_t)$$

$$\tilde{C}_t = tanh(VQC_3(v_t))$$

$$c_t = (f_t * c_{t-1}) + (i_t * \tilde{C}_t)$$

$$o_t = \sigma(VQC_4(v_t))$$

$$h_t = VQC_5(o_t * tanh(c_t))$$

$$y_t = VQC_6(o_t * tanh(c_t))$$

QLSTM uses Variational Quantum Circuits (VQCs) — quantum circuits with gate parameters optimized (or trained) that replace classical neural networks in LSTM cells, serving as feature extractors and compressors [18]. These quantum cells employ quantum entanglement to facilitate better storage and retrieval of sequential information. Every circuit block used in a QLSTM cell consist of three layers: the Forget Block, the Input and Update Block, and the Update Block.

To construct a QLSTM cell, the aforementioned VQC blocks are stacked together.

The QLSTM model has 4 variable hyperparameters - 1) Number of epochs, 2) Batch size, 3) Learning rate, and 4) Number of qubits.

B. Particle Swarm Optimisation (PSO)

Particle Swarm Optimisation is a bio-inspired metaheuristic optimization algorithm based on the behavior of a fish or bird swarm in nature. It aims to search and find the optimal solution in a multidimensional search space by simulating the pattern and movement of particles [19]. Like other metaheuristics, PSO begins with the initialization of particles with velocity and position set arbitrarily. Each particle is a representation of a solution. The particle positions and velocities are updated iteratively based on a fitness function and a global best solution found by the swarm. PSO involves the movement of particles through a search space based on mathematical calculations. The update equations for the particle's velocity (V_i) and position (X_i) in a typical PSO algorithm are expressed as follows:

The velocity update equation is given by:

$$V_i(t+1) = w \cdot V_i(t) + c_1 \cdot r_1 \cdot (P_i(t) - X_i(t)) + c_2 \cdot r_2 \cdot (G(t) - X_i(t)) \quad (1)$$

The pseudocode for PSO is given below :

Algorithm 1: Particle Swarm Optimization (PSO)

---

1 Initialize particles' positions and velocities in the search space;
2 **for** $iteration = 1$ *to* $MaxIter$ **do**
3   **for** *each particle* i **do**
4   Evaluate fitness function $f(x_i)$ for particle i;
5   **if** $f(x_i)$ *is better than the best fitness value of particle* i **then**
6   Update personal best position: $pbest[i] = x_i$;
7   Update personal best fitness value: $pbest\_value[i] = f(x_i)$;
8   Update global best particle: $gbest = $ particle with the best fitness among all particles;
9   **for** *each particle* i **do**
10   Update particle velocity and position using equations:;
11       $velocity[i] = inertia \times velocity[i] + cognitive\_coefficient \times random() \times (pbest[i] - position[i]) + social\_coefficient \times random() \times (best - position[i])$;
12       $position[i] = position[i] + velocity[i]$;
13 **return** $gbest$;

---

where: - $V_i(t+1)$ is the updated velocity of particle i at time $t + 1$, - w is the inertia weight, - c_1 and c_2 are the cognitive

and social coefficients, respectively, - r_1 and r_2 are random values between 0 and 1, - $P_i(t)$ is the best-known position of particle i at time t, - $G(t)$ is the best-known position of the entire swarm at time t, and - $X_i(t)$ is the current position of particle i at time t. The position update equation is given by:

$$X_i(t+1) = X_i(t) + V_i(t+1) \qquad (2)$$

These equations govern the movement of particles in the PSO algorithm, facilitating the exploration and exploitation of the search space to find optimal solutions. Adjusting the parameters like inertia weight (w), cognitive and social coefficients (c_1 and c_2), and others can impact the algorithm's performance in different optimization scenarios.

C. Bayesian Optimisation (BO)

Bayesian optimization is an approach to optimizing objective functions that take a long time (minutes or hours) to evaluate. It is best-suited for optimization over continuous domains of less than 20 dimensions, and tolerates stochastic noise in function evaluations. The ability to optimize expensive black-box derivative-free functions makes Bayes Optimisation extremely versatile. Bayes Optimisation consists of two main components: a Bayesian statistical model for modeling the objective function, and an acquisition function for deciding where to sample next. It builds a surrogate for the objective and quantifies the uncertainty in that surrogate using a Bayesian machine learning technique, Gaussian process regression, and then uses an acquisition function defined from this surrogate to decide where to sample.

III. PROPOSED METHODOLOGY

In this section, we explicitly explain the architecture and mechanism of the two ensemble models that we have conceptualized for forecasting 24-hour temperature predictions. Our proposed ensemble learning architecture comprises of two concurrent QLSTMs base models, which are fed with the preprocessed time-series data. The sequence lengths for the QLSTM models are set at $m \in \{3, 5\}$. To optimize the forecasting accuracy of out ensemble model, we have employed a hybrid quantum genetic algorithm - particle swarm optimization approach to fine-tune the hyper-parameter settings before actually training the QLSTM.

In the following subsections, we describe the two distinct approaches employed for QLSTM ensembling.

A. GenHybQLSTM Ensemble

In our first methodology, the application of the Quantum Genetic Algorithm - Particle Swarm Optimization (QGA-PSO) approach followed by adaptive weight ensemble technique stands as a pivotal element in elevating the performance of our QLSTM models. By employing QGA-PSO to fine-tune hyperparameters before the QLSTM training phase, we capitalize on its unique ability to explore and refine the model's configuration. This dynamic optimization process

empowers the QLSTMs to adapt more effectively to the patterns embedded within the time-series data. Consequently, this optimization step significantly enhances the accuracy and resilience of our ensemble forecasting model.

Following experimental studies, we carefully selected optimal values for the learning rate and batch size of each QLSTM through hyper-parameter tuning.

To optimize these hyperparameters effectively, we employ a Quantum Genetic Algorithm (QGA) encapsulating the QLSTM model. QGA employs the concept of quantum bits and their superposition states which can be represented as follows:-

$$| \varphi \rangle = \alpha \mid 0 \rangle + \beta \mid 1 \rangle \qquad (3)$$

where α and β are the complex numbers called the probability amplitude of corresponding state of qubit and satisfy the condition $\mid \alpha^2 \mid + \mid \beta^2 \mid = 1$. The QGA uses quantum bits and superposition states, where qubits exist in the states of $\mid 0 \rangle$, $\mid 1 \rangle$, and superposition. Quantum genetic algorithm applies the probability amplitude of qubits to encode chromosome and uses quantum rotating gates to realize chromosomal updated operation [20]. The genetic manipulation of quantum genetic algorithm is mainly through acting on the superposition state or entanglement state by the Quantum rotating gates to change the probability amplitude. The Quantum Rotating gate can been defined as given below:

$$U(\theta_i) = \begin{bmatrix} cos\theta_i & -sin\theta_i \\ sin\theta_i & cos\theta_i \end{bmatrix} \qquad (4)$$

Probability amplitudes of qubits encode chromosomes, and quantum rotating gates facilitate chromosomal updates. The pseudocode for QGA is given below: The QGA converges

---

Algorithm 2: Quantum Genetic Algorithm (QGA)

1 Initialize the quantum individuals in the population;
2 **for** $iteration = 1$ to $MaxGeneration$ **do**
3   **for** *each individual i in popSize* **do**
4   Mutate each individual of population;
5   **for** *iteration=1 to genomeLength* **do**
6   Mutate each qubit in an individual's genome;
7   Update the population with the mutation;
8   **for** *each individual in population* **do**
9   Evaluate fitness using the objective function;
10   Select the best offsprings for the next generation;
11 Update global fittest individual: $gfittest =$ individual of the particular generation evaluated to be fittest;
12 **return** $gfittest$;

---

to an optimal set of hyperparameters by minimizing the validation loss, enhancing the QLSTM's ability to capture and study the patterns in the weather dataset. The QGA-optimized
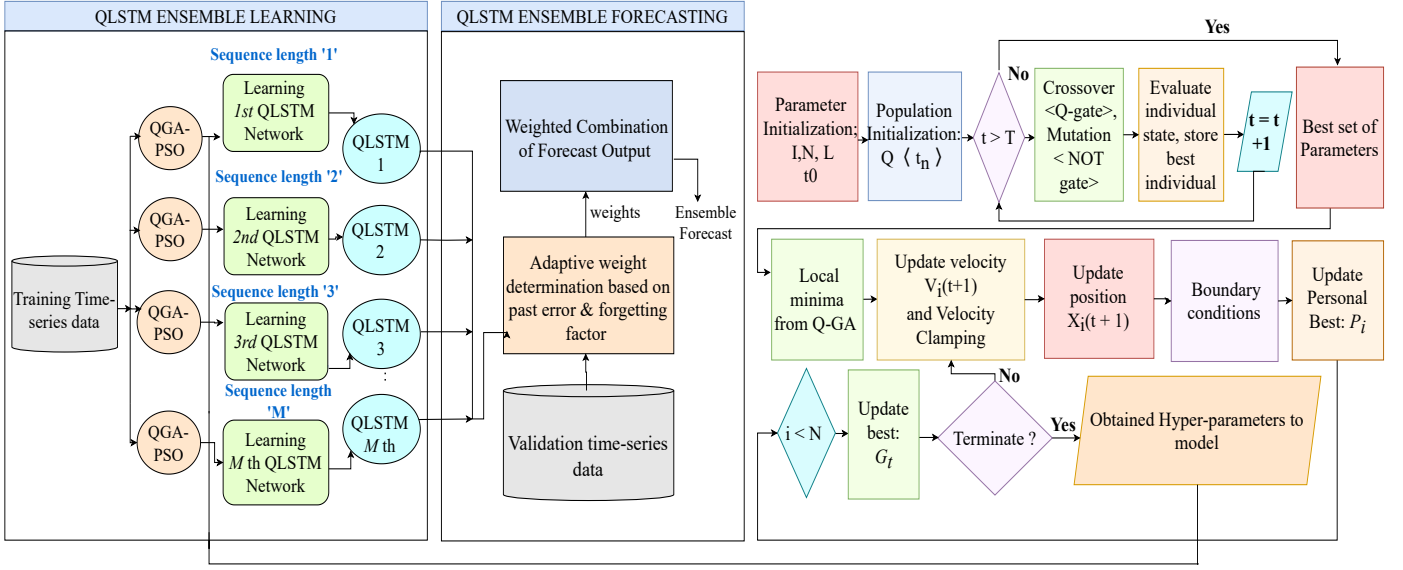
Fig. 2: Quantum Genetic-Particle Swarm Algorithm Based QLSTM network ensemble architecture (GenHybQLSTM Ensemble)

hyperparameters serve as the initial solution for further refinement using the Particle Swarm Optimization (PSO) algorithm. PSO, a bio-inspired optimization algorithm, simulates the movement of particles in a multidimensional search space. The velocity and position of particles are iteratively updated based on a fitness function and the global best solution found by the swarm. This iterative process in PSO facilitates the exploration and exploitation of the search space to find the optimal set of hyperparameters.

PSO involves the movement of particles through a search space based on mathematical calculations. The update equations for the particle's velocity (V_i) and position (X_i) in a The velocity update equation is given by:

$$V_i(t+1) = w \cdot V_i(t) + c_1 \cdot r_1 \cdot (P_i(t) - X_i(t)) + c_2 \cdot r_2 \cdot (G(t) - X_i(t))$$
$$(5)$$

These equations govern the movement of particles in the PSO algorithm, facilitating the exploration and exploitation of the search space to find optimal solutions. The hybrid

Fig. 3: Flowchart of HybQLSTM Ensemble

QGA-PSO approach enhances the efficiency of the QLSTM model by utilizing hyperparameters derived from QGA as

the initial parameters for PSO, thereby fostering expedited convergence. This collaborative approach yields finely tuned hyperparameters, enhancing the QLSTM model's performance in predicting weather attributes, as demonstrated in subsequent experimental results.

Once the best-suited values for these two critical parameters were identified, we proceeded to train the models and construct the ensemble concurrently. The aggregation mechanism to combine the predictions from the base QLSTM models is an adaptive weight scheme that accounts for the time varying dynamics of the underlying time series pattern [21]. The combining weights are computed by:

$$w_m^{(k+1)} = w_m^{(k)} + \lambda \Delta w_m^{(k)} \quad \text{for } m = 1, \ldots, M \quad (6)$$

where we have taken $\lambda = 0.85$. The $\Delta w_m^{(k)}$ is computed based on the inverse prediction error of the respective QLSTM base model as given below:

$$\Delta w_m^{(k)} = \frac{1/\varepsilon_m^{(k)}}{1/\varepsilon_m^{(1)} + 1/\varepsilon_m^{(2)} + \cdots + 1/\varepsilon_m^{(k)}} \quad (7)$$

The $\varepsilon_m^{(k)}$ is related to past prediction error measured up to the th time step in the following way:

$$\varepsilon_m^{(k)} = \sum_{t=k-\nu+1}^{k} \gamma^{k-t} e_m^{(t)} \quad (8)$$

where $0 < \gamma \leq 1, 1 \leq \nu \leq k$ and $e_m^{(t)}$ is the prediction error at each time step of the mth QLSTM model. Here, we have taken the forgetting factor $\gamma = 0.85$. Utilizing the equations (3), (4), and (5), the weights are updated over time series with
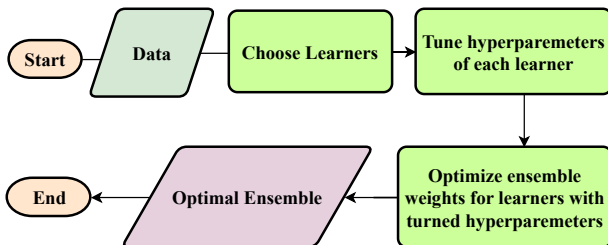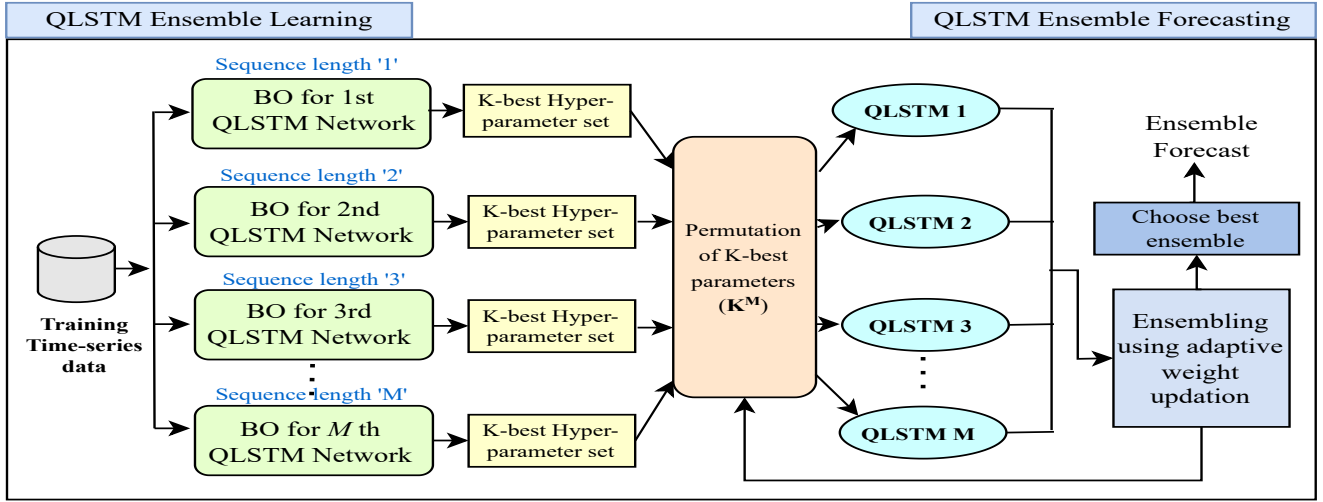
Fig. 4: Bayesian based nested optimised QLSTM network ensemble architecture (BO-QEnsemble)

T time steps $(k = 1, \cdots, M)$, leading to the final weights $w_m^{(T)}(m = 1, \cdots, M)$. Finally, the weight to be assigned to each LSTM model in an ensemble is computed as follows:

$$w_m = \frac{w_m^{(l)}}{\sum_{n=1}^{M} w_n^{(T)}} \quad \text{for } m = 1, \dots, M \quad (9)$$

the weights computed using (6) satisfy the constraints $0 \leq w_m \leq 1$ and $\sum_{m=1}^{M} w_m = 1$.

Updating weights based on previous errors, with a forgetting factor, gives more importance to recent errors, improving our model's adaptability.

B. BO-QEnsemble Architecture

In our pursuit of enhancing forecasting accuracy for 24-hour temperature predictions, we introduce a novel ensemble model— the second architecture in our framework. This innovative approach combines the power of Quantum Long Short-Term Memory (QLSTM) base models with the efficiency of Bayesian optimization for hyperparameter tuning. At the heart of this model lies an ensemble comprising m QLSTM base models, each coupled with a dedicated bayesian optimizer. The essence of this ensemble lies in the generation of a set of K-best hyperparameter configurations (denoted as K_m) from each Bayesian optimizer. These configurations collectively form a hyperparameter space, systematically permuted and applied iteratively to refine the succeeding QLSTM base models within the ensemble.

To elaborate on this approach, let's consider the process:

For hyperparameter tuning, each QLSTM base model, distinguished by its sequence length, is equipped with a bespoke Bayesian optimizer. Bayesian optimization (BO) assumes a pivotal role in locating optimal hyperparameters by minimizing the objective function within a specified domain. Each Bayesian optimizer independently produces a set of K-best

hyperparameter configurations (K_m). The amalgamation of these sets forms the complete hyperparameter space through systematic permutations.(10).

$$x^* = \arg\min_{x \in X} f(x) \quad (10)$$

where $f(x)$ denotes the score to be minimized, representing the loss function of the respective QLSTM models, X is the domain of the hyperparameter values, and x^* is a combination of hyperparameters that produces the lowest value of the score $f(x)$. Bayesian optimizers independently generate a set
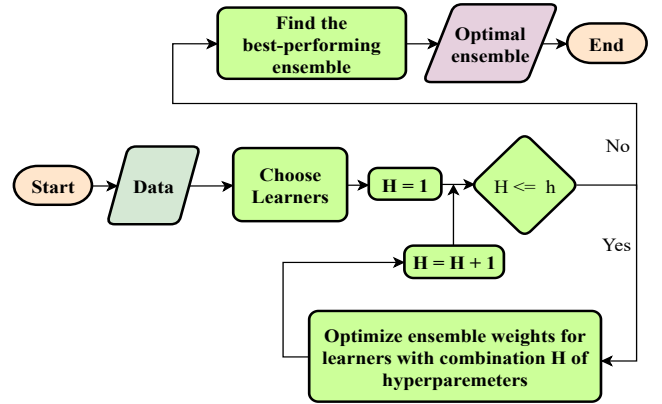

Fig. 5: Flowchart of BO-QEnsemble

of K-best hyperparameter configurations (K_m). The entire hyperparameter space is formed by permuting these sets. The hyperparameter space is used to fine-tune subsequent QLSTM base models iteratively. Predictions from each base model are obtained post-tuning. Then we use the adaptive weighted method for ensemble prediction from these trained QLSTM models. The adaptive weighting equation is defined as follows:

$$w_m^{(k+1)} = w_m^{(k)} + \lambda \Delta w_m^{(k)} \quad \text{for } m = 1, \dots, M \quad (11)$$

This method takes into account past errors and incorporates a forgetting factor. From the K-best parameter sets generated for m models, we can create K^m permutations, resulting in k^m distinct ensemble models. The weight calculation method is kept same as the previous method (equation no. (6), (7), (8) and (9)).

The algorithm for the second ensemble model approach is outlined as follows:

Inputs:
- Data set $D = \{(x, y) : x \in \mathbb{R}^{n \times p}, y \in \mathbb{R}\}$
- m base learning algorithm
- Hyperparameter sets $h_1, \ldots, h_m$
- Bayesian search chooses k hyperparameter combinations for each learner

---

Algorithm 3: Ensemble Model Training with Bayesian Optimization

---

1 **for** $HyperParamCombination = 1, \ldots, b^m$ **do**
2   **for** $model = 1, \ldots, m$ **do**
3   **for** $i = 1, \ldots, n$ **do**
4   Split D into $D_{i,\text{train}}, D_{i,\text{test}}$ for the ith split;
5   Train base learner i with hyperparameter combination on $D_{i,\text{train}}$;
6        $P_{i,model}$: Predict on $D_{i,\text{test}}$;
7      $error_{model} = (P_{1,model}, \ldots, P_{m,model})$
8   Use $error_{model}$ to compute $weights$;
9   Calculate optimal objective value (MSE), optimal weights $(W_{1H}, \ldots, W_{mH})$, and ensemble predictions;
10 Find the minimum of objective values (MSE);
11 Find the optimal weights $W_{1H}, \ldots, W_{mH}$ corresponding to the minimum objective value;

Outputs:
- Optimal objective value (MSE*)
- Optimal combination of hyperparameters $h_1, h_2, \ldots, h_m$
- Optimal ensemble weights $w_1, \ldots, w_m$
- Prediction vector of ensemble with optimal weights

Experimentation with this model architecture has demonstrated a noticeably improved performance compared to the GenHybQLSTM model. This enhancement is likely attributed to the systematic permutation of hyperparameters for each individual model. A detailed study of the results from both models is done in the Results and Discussion section of this paper where we have compared with other state of the art models.

IV. EXPERIMENT DETAILS

A. Dataset Description

The dataset utilized in this research was created by scraping from the official website of the Government of Canada weather-related data for the region of Ottawa from January 1st, 2010 to December 31st, 2020 [22] [16]. It has the following features: date, time (in 24 hours), temperature, dew point temperature, relative humidity, wind speed, visibility (in kilometers), Pressure (in kilopascals), and precipitation amounts (in millimeters). It also had a few derived features like humidity index and wind chill, which we did not take into account to keep our list of features as independent as possible from each other. The compiled data set comprised 96,432 rows of data for 8 variables, where each row represents an hour.

B. Dataset Preprocessing

The raw scraped dataset underwent a preprocessing routine wherein the missing data points were imputed using the median values of the respective feature columns. Subsequently the features were scaled using the RobustScaler standardization procedure. In this the median is removed and the data is scaled according to the Inter- Quartile Range (IQR). The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile). This was done to scale the features in such a way that it is robust to outliers. Following this, 87% of the 96,432 rows was allocated for training purpose and the remaining 13% for model testing, to ensure a balanced representation in both the training and testing subsets. Subsequently, Z-score normalization was implemented to ensure a homogeneous distribution of feature scales across the dataset. Following this the values of each feature was transformed such that they have a mean of 0 and a standard deviation of 1.

C. Experimental Setup

The experiments of this work are implemented in Python 3.10.11 and utilizes the libraries such as Numpy 1.24.3, Pandas 2.0.3, PyTorch 2.1.1 and Matplotlib. Quantum Machine Learning Library - PennyLane 0.34.0 was pivotal in integrating quantum circuits and functionalities into the models.

V. RESULTS AND DISCUSSION

In this section, we present the outcomes of our experimental evaluation, analyzing the comparative performance of various models employing distinct optimization algorithms. In the course of our investigation, we have proposed our novel methodologies, GenHybQLSTM and BO-QEnsemble and compared it against other state-of-the-art models. StandardScaler was employed to enhance the convergence and stability in seasonal data training, averting dominant feature emergence. Mean Squared Error serves to minimize training loss. Our primary goal is to enhance forecasting and accelerate convergence, achieved through the integration of meta-heuristics. We introduce a hybrid quantum algorithm for QLSTM hyper-parameter tuning, addressing issues in conventional quantum genetic algorithms, such as premature convergence and limited local search ability. To overcome these limitations without compromising model performance, we incorporate a hybrid QGA-PSO parameter optimization.

This approach mitigates premature loss of diversity in the population, often encountered with fitness-driven evaluations, and enhances local search performance, ensuring a more expedited and effective convergence.

Furthermore, the employment of the Bayesian Optimization algorithm within the BO-QEnsemble model, substantiated by commendable results, attests to its efficacy in ascertaining the optimal hyper-parameter setting. Proficient in global optimization, it navigates complex spaces, specifically calibrated for temporal trend prediction. Employing a probabilistic surrogate model, it provides predictions and quantifies uncertainty, enhancing BO-QEnsemble performance in the QLSTM ensemble design.

TABLE I: Hyperparameter sets and ensemble weights for QLSTM's

Parameter	GenHybQLSTM	BO-QEnsemble
Learning Rate	[0.0677, 0.0691]	[0.0726, 0.0522]
Layers	[2, 3]	[2, 2]
Number of Qubits	[6, 6]	[2, 4]
Hidden Units	[5, 5]	[5, 4]
Sequence Length	[3, 5]	[3, 5]
Batch size	[66, 160]	[64,224]
Combining Weight	[0.50272, 0.49728]	[0.50123, 0.49876]

The choice of Quantum LSTM over classical LSTM as the foundational models for our ensemble architecture significantly contributes to the observed performance enhancement. QLSTM demonstrates accelerated learning, extracting more information immediately after the first training epoch, with a more pronounced and less volatile decrease in both train and test loss compared to the classical LSTM counterpart. Lever-
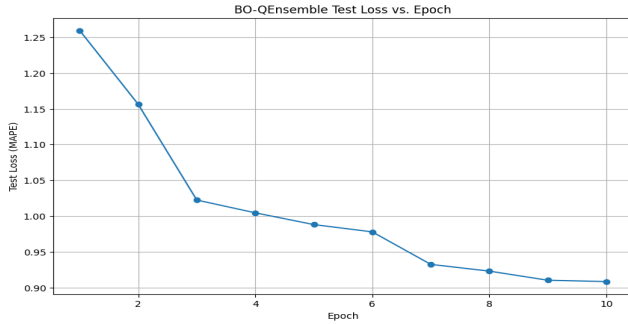

Fig. 6: Test loss Vs Epoch

aging quantum circuits modeled after quantum computing phenomena, QLSTM exhibits faster convergence and superior identification of local features, such as minima and maxima, particularly advantageous when confronted with complex temporal structures within the input sequential data. The test loss vs epoch graph for best performing BO-QEnsemble model is shown in Fig 6.

Notably, the results underscore the superior efficacy of the BO-QEnsemble, which consistently outperforms alternative

models by a substantial margin. Specifically, BO-QEnsemble attains the most favorable outcome with a Mean Absolute Percentage Error (MAPE) of 0.91, surpassing GenHybQLSTM (MAPE: 0.92) and QLSTM (MAPE: 1.12). A comprehensive summary of MAPE of these models is provided in Table 2. All reported values are aggregated over ten independent statistical trial runs, and the figures presented are denominated in standardized units. Additionally, Fig 7 presents the forecast

TABLE II: Comparision of mean absolute percentage error (mape)

Models	MAPE
Quantum LSTM	1.12
LSTM	1.99
Proposed BO-QEnsemble	0.91
Proposed GenHybQLSTM	0.92
DE-ANN	1.15
Artificial Neural Network	2.09
Gated Recurrent Unit (GRU)	1.98

plot of the BO-QEnsemble model, showing its predictive insights, while Fig 8 displays the forecast plot for the GenHybQLSTM model. While quantum LSTM shows only a slight
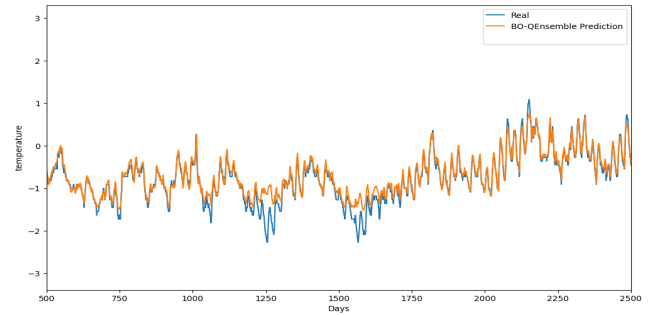

Fig. 7: Forecast Plot for the best performing BO-QEnsemble Model

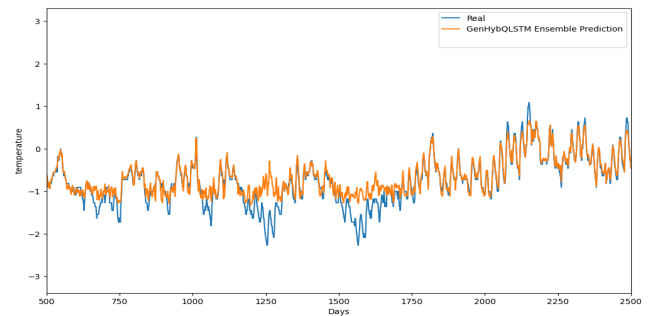improvement in MAPE over DE-ANN, its computational

Fig. 8: Forecast Plot for the GenHybQLSTM Ensemble Model

efficiency outperforms DE-ANN, which is hindered by the complexity of the differential evolution algorithm, resulting in slow performance. Ensemble models consistently outperform

individual QLSTM, LSTM, GRU, or ANN models, evident in lower MAPE. This approach enhances generalization, mitigates overfitting, and improves overall performance on test data by capturing complex temporal patterns during training, reducing variance, and aligning predictions with a realistic range. Figure 9 shows the 24-hour forecast plot obtained from the best performing BO-QEnsemble model.
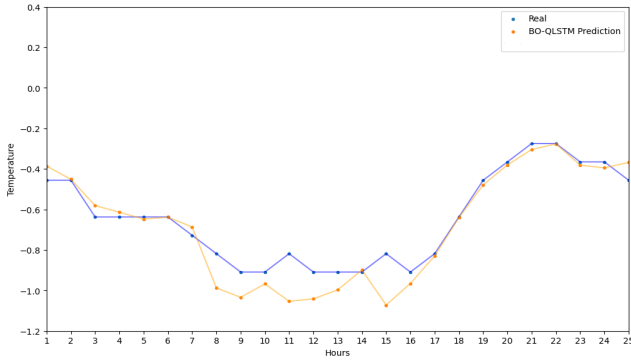

Fig. 9: 24-hour Forecast Plot for short term testing of the best performing BO-QEnsemble Model

Hence it is conclusive that the proposed ensemble architectures perform better than individual QLSTM networks across a wide range of performance metrics and between the two ensembles the proposed BO-QEnsemble model is superior to the GenHybQLSTM model.

VI. CONCLUSION AND FUTURE WORK

This manuscript introduces two novel hybrid ensemble models BO-QEnsemble & GenHybQLSTM arcitectures and a QGA-PSO optimised QLSTM network. These has been compared to other classical deep learning models developed for the forecasting of weather over a speicified geographical area. Experimental results showcase that our BO-QEnsemble Architecture stands as a SOTA benchmark for the given weather forecasting task and demonstrates superior results in comparision to other models in terms of performance metrics such as mean absolute percentage error (MAPE) and mean squared error (MSE). Future research may be directed towards the incorporation of the proposed ensemble models for testing in different geographical areas and as well as apply it in different application domains.

REFERENCES

[1] S. Ziyabari, L. Du, and S. Biswas, "A spatio-temporal hybrid deep learning architecture for short-term solar irradiance forecasting," in *2020 47th IEEE Photovoltaic Specialists Conference (PVSC)*, pp. 0833–0838, 2020.

[2] M. A. R. Suleman and S. Shridevi, "Short-term weather forecasting using spatial feature attention based lstm model," *IEEE Access*, vol. 10, pp. 82456–82468, 2022.

[3] A. Guin, "Travel time prediction using a seasonal autoregressive integrated moving average time series model," in *2006 IEEE Intelligent Transportation Systems Conference*, pp. 493–498, 2006.

[4] G. D. Silvestre, M. R. dos Santos, and A. C. de Carvalho, "Seasonal-trend decomposition based on loess + machine learning: Hybrid forecasting for monthly univariate time series," in *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, 2021.

[5] H. Cui, D. Huang, Y. Fang, L. Liu, and C. Huang, "Webshell detection based on random forest–gradient boosting decision tree algorithm," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pp. 153–160, 2018.

[6] A. Zhukov, D. Sidorov, A. Mylnikova, and Y. Yasyukevich, "Random forest, support vector regression and gradient boosting methods for ionosphere total electron content nowcasting problem at mid-latitudes," in *2018 2nd URSI Atlantic Radio Science Meeting (AT-RASC)*, pp. 1–3, 2018.

[7] H. Al-Shehri, A. Al-Qarni, L. Al-Saati, A. Batoaq, H. Badukhen, S. Alrashed, J. Alhiyafi, and S. O. Olatunji, "Student performance prediction using support vector machine and k-nearest neighbor," in *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1–4, 2017.

[8] T. Guo, R. Liu, H. Yang, L. Shi, F. Li, L. Zhang, Y. Chen, Z. Liu, and F. Luo, "Predict atmosphere electric field value with the lstm neural network," in *2017 International Conference on Computer Systems, Electronics and Control (ICCSEC)*, pp. 263–266, 2017.

[9] B. Ren, X. Xu, and H. Yu, "Research of lstm-rnn model and its application evaluation on agricultural products circulation," in *2021 IEEE 3rd Eurasia Conference on IOT, Communication and Engineering (ECICE)*, pp. 467–471, 2021.

[10] F. X. et al., "A short-term load forecasting model based on neural network considering weather features," in *2021 IEEE 4th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, pp. 24–27, 2021.

[11] A. B. S. H. T. Wardah, A. A. Kamil and W. W. I. Maisarah, "Statistical verification of numerical weather prediction models for quantitative precipitation forecast," in *2011 IEEE Colloquium on Humanities, Science and Engineering*, pp. 88–92, 2011.

[12] I. H. G. A. M. Abdalla and A. A. Tamimi, "Deep learning weather forecasting techniques: Literature survey," in *2021 International Conference on Information Technology (ICIT)*, pp. 622–626, 2021.

[13] N. Alshammari and J. Asumadu, "Comparative analysis of meta-heuristic algorithms for optimal sizing of hybrid renewable energy system," in *2020 IEEE International Conference on Electro Information Technology (EIT)*, pp. 648–654, 2020.

[14] B. Nath Saha and A. Senapati, "Long short term memory (lstm) based deep learning for sentiment analysis of english and spanish data," in *2020 International Conference on Computational Performance Evaluation (ComPE)*, pp. 442–446.

[15] A. Candelieri, "A gentle introduction to bayesian optimization," in *2021 Winter Simulation Conference (WSC)*, pp. 1–16, 2021.

[16] A. Sen, A. Rhik Mazumder, D. Dutta, U. Sen, P. Syam, and S. Dhar, "Comparative Evaluation of Metaheuristic Algorithms for Hyperparameter Selection in Short-Term Weather Forecasting," *arXiv e-prints*, p. arXiv:2309.02600, Sept. 2023.

[17] R. A. Viswambaran, G. Chen, B. Xue, and M. Nekooei, "Evolutionary design of long short term memory (lstm) ensemble," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 2692–2698, 2020.

[18] S. Y.-C. Chen, S. Yoo, and Y.-L. L. Fang, "Quantum long short-term memory," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[19] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, pp. 1942–1948 vol.4, 1995.

[20] Y.-k. Zhang, J.-c. Liu, Y.-a. Cui, X.-h. Hei, and M.-h. Zhang, "An improved quantum genetic algorithm for test suite reduction," in *2011 IEEE International Conference on Computer Science and Automation Engineering*, vol. 2, pp. 149–153, 2011.

[21] J. Y. Choi and B. Lee, "Combining lstm network ensemble via adaptive weighting for improved time series forecasting," *Mathematical Problems in Engineering*, vol. 2018, p. 2470171, 2018.

[22] A. R. Mazumder, "Harvard Dataverse, Ontario Weather ," 2023.