Air Quality, Atmosphere & Health (2023) 16:1871–1890
https://doi.org/10.1007/s11869-023-01380-7

Air quality index prediction based on three-stage feature engineering, model matching, and optimized ensemble

Yucheng Yin[1] · **Hui Liu**[1]

Received: 26 November 2022 / Accepted: 15 May 2023 / Published online: 23 May 2023

Abstract
A prompt and accurate prediction of air quality index (AQI) has become a necessity to tackle the mounting environmental threats. This paper proposes a feature-driven hybrid method for hourly, 3-step-ahead, and deterministic AQI prediction, which includes three modules. In Module 1, an "extract-merge-filter" procedure of feature engineering is created to capture the potential features from the AQI series. Ten feature sets are generated as candidates. In Module 2, six models including Light Gradient Boosting Machine, Extreme Gradient Boosting, Long Short-Term Memory, Convolutional Neural Network, Multilayer Perceptron, and Deep Neural Network are developed as base predictors and performed on the candidate features. In Module 3, predictors are first matched with their optimal features using a comprehensive metric, and then combined in an optimized ensemble using OPTUNA. A case study on the AQI data from four different Chinese cities is carried out to demonstrate the method. The experimental results show the following: (1) Feature engineering significantly boosts prediction performance and provides interpretable findings for practical use. (2) Customized input of features to the predictors is more effective than a fixed input and can rise the performance to a higher level. (3) OPTUNA is a promising tool for optimizing ensemble weights. The final ensemble model is superior to single machine learning models and has a good robustness.

Keywords AQI prediction · Feature engineering · Model matching · Optimized ensemble · OPTUNA

Introduction

Over the past decades, the intensive agricultural and industrial activities, and the constant growth of population have escalated the pollutant emissions in the air. Numerous studies have stated that poor air quality is one of the biggest threats to both ecosystems and human existence (Wu et al. 2007; Guan et al. 2016; He et al. 2019; Guo et al. 2020). In this regard, close monitor and accurate and timely prediction of atmospheric pollution have become essential. The government planners can take precautionary measures and issue early warnings and citizens can be guided towards healthier travel choices (Yang et al. 2017).

The air quality index (AQI) is a widely used indicator that comprehensively reports the status of ambient air quality. It measures the concentrations of six major air pollutants—carbon monoxide (CO), sulfur dioxide (SO_2), nitrogen dioxide (NO_2), ozone (O_3), fine particulate matter ($PM_{2.5}$), and inhalable particulate matter (PM_{10})—and converts them into a unified dimensionless indicator (Liu 2002; Kim et al. 2022). AQI levels offer clear and concise advice on how to protect oneself from air pollution. For instance, healthy individuals can continue with normal activities or reduce outdoor exposure, while vulnerable individuals may need to take more rigorous protective measures. AQI values are usually calculated and published at intervals of hours, days, and months. The AQI time series exhibits four characteristics, including autocorrelation, periodicity, non-stationarity, and nonlinearity. (a) Autocorrelation refers to the tendency of the series to exhibit correlation between successive observations, indicating that current values are dependent on past values. This correlation gradually weakens over time. In most studies, the recent observations of the predicted target are generally used as the only input to the model (Zhang et al. 2020; Surakhi et al. 2021). (b) Periodicity refers to the

✉ Hui Liu
csuliuhui@csu.edu.cn

[1] Institute of Artificial Intelligence and Robotics (IAIR), Key Laboratory of Traffic Safety On Track of Ministry of Education, School of Traffic and Transportation Engineering, Central South University, Changsha 410075, Hunan, China

presence of recurring patterns, such as daily, weekly, and seasonal cycles, that are commonly observed in AQI data. For example, significant differences have been found in the distribution and variation patterns of AQI values between winter and summer (Wu et al. 2017). (c) Non-stationarity and nonlinearity refer to the time-varying nature of the mean, variance, and other statistical properties of the series. They arise from the complex interactions between multiple pollution sources and meteorological factors (Bitencourt et al. 2022).

Understanding and modeling these characteristics is crucial for air pollution prediction. Various techniques, including physical, statistical, and machine learning (ML) models, have been explored (Ojagh et al. 2021). Physical models simulate and analyze meteorological, aerodynamic, and chemical reaction mechanisms for air pollutant generation, accumulation, and diffusion. Numerical Weather Prediction (NWP) models are a representative example (Zhou et al. 2017), but they have limited practicality for short-term predictions due to latency issues (Liu and Chen 2019). In contrast, statistical models, such as Autoregressive Integrated Moving Average (ARIMA) models, make predictions solely based on statistical data relationships without considering the chemical and physical evolution mechanisms of pollutants (Zhang et al. 2018). However, these models cannot accurately predict significant changes in non-stationary series, resulting in a loss of accuracy. On the other hand, ML models have gained recognition for their excellent nonlinear fitting ability, generalization, and real-time performance (Liu et al. 2021b). In the field of air quality prediction using ML models, two main tasks need to be addressed. First, mining the information that describes autocorrelation, periodicity, and other characteristics from the non-stationary original time series. Second, building a prediction system with high accuracy and stability.

A broad consensus in the industry has been that the upper limit of machine learning is controlled by data and features, and what the models do is just to try to approach this limit (He et al. 2021). Effective feature engineering boils down to a deep understanding of the dataset and has been of paramount importance. Luo et al. (2019) provided their winning solution in 2018's KDD CUP of Fresh Air; the task is to predict the air quality of the next 48 h in Beijing and London. They designed six groups of features. For the air quality data, statistical features such as the mean, median, maximum, and minimum values over different windows in the past 1 to 72 h were derived from the series. These feature groups, in proper combination, greatly improved the performance. More elaborate works were done by Masmoudi et al. (2020). They extracted datetime features (such as hour of the day, day, month, and season) and sliding window features (such as 24 h, 48 h, 169 h lagged rolling mean concentration of pollutants) and adopted the Random Forest (RF) to estimate feature quality and remove bad ones. Experiments showed that feature selection can narrow the training down to the most contributing features and improve algorithmic speed, accuracy, and processor usage. Another type of approach that has been proposed more frequently is based on the exceptional capacities of the certain neural networks as feature extractors. For instance, temporal and spatial features were extracted and dimensionally reduced by Convolutional Neural Network (CNN). Long Short-Term Memory (LSTM) network then processed the multivariate outputs and learned the serial dependencies (Li et al. 2020; Dai et al. 2021; Elmaz et al. 2021).

In terms of prediction systems, Decision Trees (DTs) and Artificial Neural Networks (ANNs) are two common solutions to a time series forecasting problem with machine learning (Jamei et al. 2022). The Decision Tree model is based on a tree structure, heuristically selects features to partition the feature space, and finally makes decisions in the leaf nodes of each partition. Thongthammachart et al. (2022) found that Light Gradient Boosting Machine (LightGBM) was superior to RF and Extreme Gradient Boosting (XGBoost) in estimating NO_2 and $PM_{2.5}$ concentrations, especially at high concentrations, which shed light on the risk assessment. Moreover, ANNs have become promising tools in recent years because of their strengths in nonlinear expression (de Gennaro et al. 2013). Multiple studies that employ ANNs in atmospheric pollution predictions have been done. The representative models include Multilayer Perceptron (MLP) (Li and Jin 2018; Perez and Menares 2018), Elman Neural Network (ENN) (Hao and Tian 2019), LSTM (Liu et al. 2021a; Kim et al. 2022), and Deep Neural Network (DNN) (Eslami et al. 2020). All these models are solid approximators, though the performance can rise to a higher level through sophisticated methods like ensemble learning (Singh et al. 2013; Li et al. 2021). Within the surveyed studies, for example, Liu et al. (2019) constructed a hybrid model for hourly fine particle concentrations prediction, in which four ENNs with different architectures were set as base learners and the output prediction results were stacked on another model (also called meta-learner), Outlier-Robustness Extreme Learning Machine (ORELM), to make the final predictions. The proposed model was shown to outperform individual learners. The other popular ensemble method is weight-based. Base learners are separately trained, and then their prediction results are assigned with ensemble weights. Aiming at the calibration of ensemble weights, other applicable techniques such as Particle Swarm Optimization (PSO) (Zhu et al. 2018), Multi-Objective Wolf Colony Algorithm (MOWCA) (Liu and Yang 2021), and Q-learning (Liu et al. 2021c) are commonly used.

Still, the existing studies are expected to be further explored in feature engineering, model matching, and ensemble optimization: (a) In terms of feature engineering,

despite the reported distinct increase in the accuracy and robustness, the network-based methods remain black-boxes for humans. They cannot tell the users what features are promising and what are not. In this sense, series-derived feature engineering is more preferred and interpretable. (b) In terms of the model matching, most studies placed emphasis on the heterogeneity of the base models to ensure a good ensemble, while inputting the same data or features into each model. The mismatch between data and models may occur and influence the ultimate performance. For example, ANNs perform better on continuous variables and slightly worse on discrete variables. In this sense, we are motivated to dynamically find the best features for different models. (c) In terms of ensemble optimization, some algorithms suffer from computation burden and are not lightweight enough.

To remove the aforementioned restrictions, this study proposes a hybrid AQI prediction model that involves three-stage feature engineering, dynamic matching of predictors and features, and optimized ensemble. The main contributions and innovations are summarized as follows:

- **An "extract-merge-filter" procedure of feature engineering is put forward to provide sufficient learning materials for base predictors.** In the extraction stage, diverse classes of features are derived from the raw time series to capture multi-resolution knowledge, such as trending, seasonal or cyclical factors, and irregular changes. In the merging stage, several pairs of features are combined to exploit their complementary discriminative strengths. In the filtering stage, unsatisfactory features are excluded to varying degrees by a built-in feature ranking algorithm of the LightGBM.
- **A dynamic matching between base predictors and feature sets is performed based on a comprehensive evaluation metric.** Six classical models, including tree models and ANNs, are chosen as base predictors. Regarding that they perform differently on various features, all 10 feature sets are evaluated. In terms of the performance evaluation, we design a comprehensive metric based on the idea of sorting, which integrates three mainstream evaluation criteria, including MAE, RMSE, and PCC. This metric not only avoids the weak coupling between single metrics but also makes the matching more reasonable.
- **An effective and lightweight framework called OPTUNA is utilized to optimize the ensemble weights between multi-predictors.** OPTUNA is initially designed to solve the hyperparameter optimization (HPO) problem, mostly for hyperparameters like connection weights and kernel size within the model architecture (Domashova and Mikhailina 2021; Pravin et al. 2022; Sipper and Moore 2022; Srinivas and Katarya 2022). To the best of our knowledge, there are scarce studies applying OPTUNA to ensemble learning, let alone in the field of air quality prediction. It can be a good attempt since the weights between models can also be treated as hyperparameters.

Methodology

Framework of the proposed model

AQI prediction is a complex task with different approaches depending on the input variables used. In this study, we focus on developing a univariate model that solely relies on AQI as input to make 1-h, 3-step-ahead deterministic AQI predictions. Previous studies have shown that univariate approaches are promising for short-term AQI prediction and are easier to interpret and implement in practice (Castelli et al. 2020; Ji et al. 2022; Li et al. 2022). Additionally, univariate models allow us to better understand the relationship between AQI and the selected features. The framework of the proposed model is shown in Fig. 1. It includes three modules: (1) three-stage feature engineering, (2) two-class base predictors modeling, and (3) dynamic multi-predictor ensemble. They are described as follows:

Module 1: An "extract-merge-filter" procedure of feature engineering is proposed. In the extraction stage, multiple classes of candidate features are extracted from the historical AQI time series ahead of the current time step, including timestamp features, lag features, differential features, and window features. The following two stages focus on generating feature sets from candidate features in two different ways. In the merging stage, the features are merged in classes, obtaining five feature sets. In the filtering stage, all the candidate features are input into LightGBM and sorted by importance. Another five feature sets are filtered out based on different thresholds (top 5, 10, 15, 20, 25). A total of 10 different feature sets are designed and prepared for subsequent experiments.
Module 2: Six models from tree models and ANNs are selected as base predictors. They are LightGBM, XGBoost, LSTM, CNN, MLP, and DNN. All feature sets will be trained and evaluated on these six base predictors. It not only guarantees model diversity for ensemble but also offers a chance to fully exert the strengths that models have on different feature inputs.
Module 3: The six base predictors are trained with the ten feature sets. The pair of base predictor and feature set that gives the best performance is recognized as the optimal matching. Given that adopting different individual evaluation metrics may point to different conclusions, a comprehensive metric called $rank_{all}$ is proposed. It integrates the evaluation results under the three mainstream

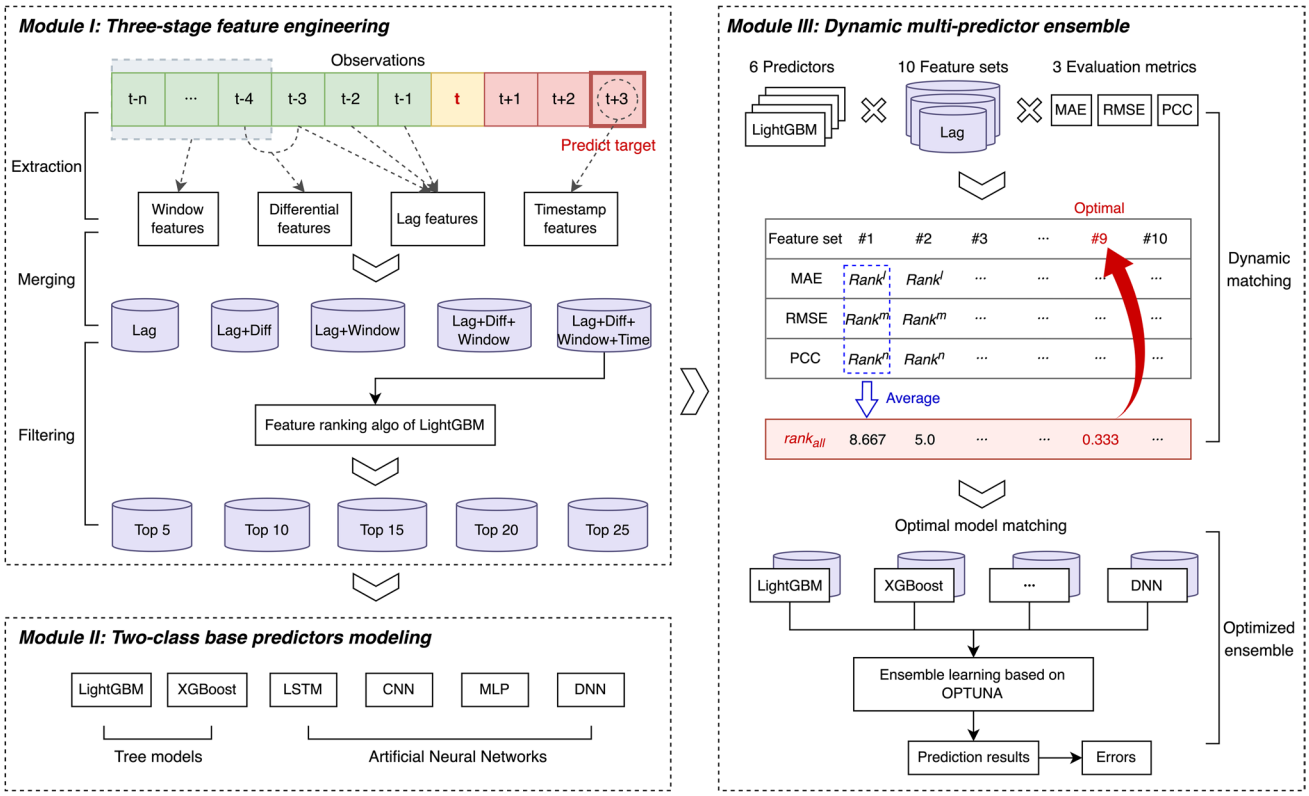

Fig. 1 Framework of the proposed model

evaluation metrics, namely MAE, RMSE, and PCC, in a sorted and averaged way. After the matching is done, the results are combined in an ensemble where the ensemble weights are optimized using the hyperparameter optimization framework OPTUNA. At this point, the final prediction results are obtained.

The proposed model is customizable and practical for use in different cities. On one hand, the dynamic model matching and ensemble are automated processes that do not require much human intervention and supervision. On the other hand, once the model training is complete, users only need to make individual predictions without the need for frequent model updates. The execution time for prediction can be very short, which does not burden actual operations. However, it is recommended to retrain the model when there is a significant change in the city environment or monitoring equipment.

Module 1: Three-stage feature engineering

Stage one: Candidate feature extraction In this stage, four classes of candidate features are extracted. Figure 2 shows a

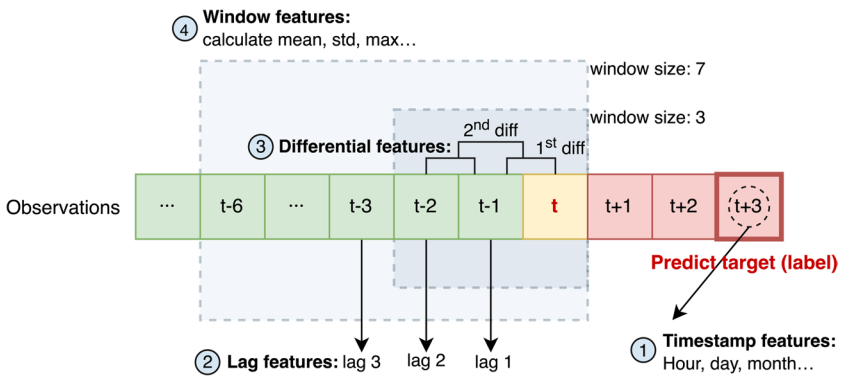

Fig. 2 Feature extraction process

simple schematic of the process of extracting these features based on a series.

- Timestamp features

Timestamp features are derived from the datetime type of variables. They can generally be subdivided into calendar features, such as year, quarter, month, week, day, and hour, and Boolean features, such as whether it is a weekend and whether it is a holiday, which are used to flag special datetime. This study looks at just 1 year of data. Features such as months and quarters that are not periodic and do not provide valuable information for model training are not retained.

- Lag features

Lag features represent past observations in a time series that measure its autocorrelation. Figure 3 presents the autocorrelation plot generated from the AQI series in Dataset #1. The plot shows the correlation between the current observation and the lag values from prior time steps, indicated by vertical lines with dots at the top. From Fig. 3, we observe that the series exhibits a long-term autocorrelation, and the correlation weakens over time. Too many lag features may lead to redundancy and overfitting. To strike a balance between capturing the essential information of the series and avoiding unnecessary complexity in the model, we focus on the most highly correlated features, with a correlation coefficient above 0.7. As a result, 9 lag features are obtained.

An expression is given in Eq. (1):

$$lag_n^t = Y[t-n] \tag{1}$$

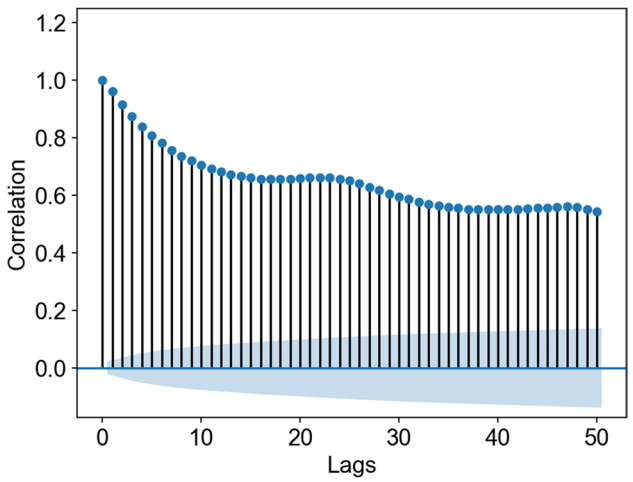

Fig. 3 Autocorrelation plot of AQI series

where n denotes the number of lagged steps, taken from 1 to 9. Y denotes the observations series. t denotes the current time step.

- Differential features

Differential features capture trend information and are obtained by shifting the time series and making a difference. For simplicity, this study only extracts first-order and second-order differential features, which are expressed in Eq. (2) and Eq. (3):

$$diff_1^t = Y[t] - Y[t-1] \tag{2}$$

$$diff_2^t = diff_1^t - diff_1^{t-1} = (Y[t] - Y[t-1]) - (Y[t-1] - Y[t-2]) \tag{3}$$

- Window features

Window features are aggregated statistics of previous observations over a fixed length of time (called "window"), and are used to capture trending, seasonal or cyclical factors, and irregular changes in the time series. The lag feature mentioned above is also a specific kind of window feature, with a window size of 1. In this study, the current time step is included in the window; the window size is respectively taken as 12 h, 2 days, 3 days, 5 days, and 10 days; and the statistics calculated in the windows include mean, standard deviation, sum, kurtosis, skewness, and maximum. An example of calculating the mean over a window size of S observations before time step t is given in Eq. (4):

$$window_{mean}^t = mean(Y[t], Y[t-1], \cdots, Y[t-S]) \tag{4}$$

Stage two: Feature merging The four classes of features have discriminative strengths and deficiencies. For instance, the lag features provide observations at specific historical time steps but lacks a description of the ups and downs in the long and short term, which is also found to be important for enhancing prediction accuracy. The window features are global statistics in different periods and do not care much about local gain changes. The timestamp features can mine the hidden information of the time step itself that the model may also care about, such as whether it is a holiday, but when used alone, they are obviously of little help in predicting future values since there are no actual observations involved.

This complementary nature between different classes of features inspires us to propose a feature merging method, expecting that the combined features may outperform any single one. From a statistical point of view, the combined features can fit the real distribution in a higher

dimensional space. Specifically, the following five sets of features are constructed: Feature set #1: lag features, Feature set #2: lag features merged with differential features, Feature set #3: lag features merged with window features, Feature set #4: lag features merged with differential features and window features, Feature set #5: lag features merged with differential features, window features, and timestamp features.

The main idea is that the lag features proved to be one of the most contributing features in predictive analysis (Surakhi et al. 2021), so they are adopted as the baseline and merged with the other four classes of features. In addition, the timestamp features are static variables and provide relatively minor information, so they are only added in Feature set #5 to play a supplementary role.

Stage three: Feature filtering Since Feature sets #1 to #5 are formed by the inter-combination between feature classes, the differences within individual features are greatly neglected. Too many feature dimensions can also easily bring about the "curse of dimensionality" and high computational complexity.

In this stage, features are filtered instead of expanded. We focus on the contribution of each feature to the prediction model, while less caring about the feature class to which it belongs. Specifically, it is implemented through the feature selection algorithm built in LightGBM, which can quantify the contribution of each feature to the prediction model as feature importance. The detailed principle is demonstrated in the "Module 2: Two-class base predictors modeling" section. The top 5, 10, 15, 20, and 25 features of the total features in descending order of feature importance are selected to build Feature sets #6, #7, #8, #9, and #10, respectively. They will be used to study the impact of feature selection as well as the size of the feature sets on the prediction performance.

So far, all 10 feature sets for further experiments have been prepared, and their composition and data overview are given in Table 1.

Module 2: Two-class base predictors modeling

In this module, we select six popular models as base predictors, including XGBoost, LightGBM, MLP, DNN, CNN, and LSTM. The schematic diagrams of them are given in Fig. 4.

Tree models

- XGBoost

XGBoost and LightGBM are both implementations of Gradient Boosting Decision Tree (GBDT). They have strong capabilities in handling missing data and feature selection, which are particularly important in time-series analysis (Xian et al. 2022; Zhao et al. 2022). One Classification and Regression Tree (CART) is learned each iteration to fit the residuals between the true values and the prediction results of previous trees. The main improvement of XGBoost to GBDT is the modification of the cost function. A regularization term is added to avoid over-fitting issues. Moreover, GBDT only uses the first-order derivatives, while XGBoost expands the cost function into Taylor's second-order series to obtain the first-order and second-order derivatives, making the convergence faster.

XGBoost grows trees in a level-wise strategy, as shown in Fig. 4(a), where leaves at the same level are split simultaneously. Based on the basic information gain of the current feature, the optimal split is selected. There are a lot of unnecessary computational efforts due to its indiscriminate traversal. Refer to Song et al. (2020) for more detailed explanations of XGBoost.

- LightGBM

LightGBM adopts a leaf-wise tree growth strategy, where the leaf with the highest information gain is split first. As shown in Fig. 4(b), the right subtree is already much deeper than the left subtree. If the gradient partition of the right subtree is still

Table 1 Summary of feature sets considered in this study

Feature set	Composition	Size
#1	Lag	9
#2	Lag + differential	11
#3	Lag + window	39
#4	Lag + differential + window	41
#5	Lag + differential + window + timestamp	46
#6	Top 5 of the total features in descending order of importance	5
#7	Top 10 of the total features in descending order of importance	10
#8	Top 15 of the total features in descending order of importance	15
#9	Top 20 of the total features in descending order of importance	20
#10	Top 25 of the total features in descending order of importance	25

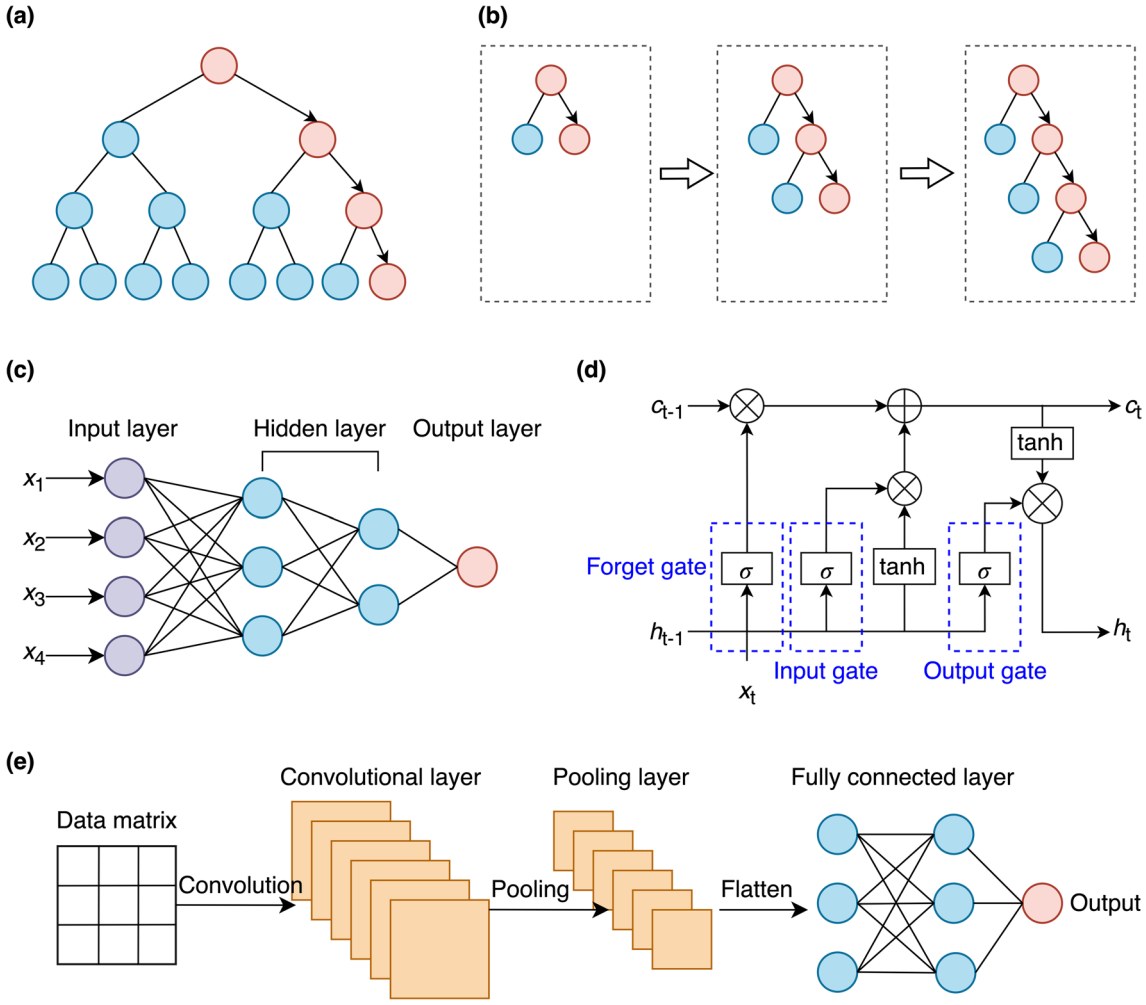

Fig. 4 Schematic diagram of the models used in this study: (**a**) XGBoost, (**b**) LightGBM, (**c**) MLP and DNN, (**d**) LSTM, (**e**) CNN

dominant over the left subtree, the right subtree will continue to be split. Compared with XGBoost, LightGBM shows faster training speed and higher prediction accuracy. The information gain of a split point (j, d) with feature j and split d is measured by the variance after splits, given in Eq. (5):

$$V(j,d) = \frac{1}{n}\left(\frac{\left(\sum_{x_{ij}\in A_l} g_i + \frac{1-a}{b}\sum_{x_{ij}\in B_l} g_i\right)^2}{n_l^j(d)} + \frac{\left(\sum_{x_{ij}\in A_r} g_i + \frac{1-a}{b}\sum_{x_{ij}\in B_r} g_i\right)^2}{n_r^j(d)}\right) \quad (5)$$

where n denotes the number of training data.x_{ij} denotes the feature j of the $_{i\text{th}}$ sample.

$$A_l = \{x_i \in B : x_{ij} \le d\}, A_r = \{x_i \in A : x_{ij} > d\},$$
$$B_l = \{x_i \in B : x_{ij} \le d\}, B_r = \{x_i \in A : x_{ij} > d\}.$$

A is the subset of first $a \times 100\%$ samples with larger gradients.

B is the subset of remaining $(1-a) \times 100\%$ samples with smaller gradients.$b \times 100\%$ is the proportion of random sampling in subset B.

Coefficient $\frac{1-a}{b}$ is used to appropriately amplify the sampled data with small gradients to maintain the distribution of the original data.g_i denotes the gradient of the $_{i\text{th}}$ sample.n_l^j and n_r^j are the numbers of the samples split into left and right child node.

For each feature j, the best split is selected by $d_j^* = \underset{D}{\text{argmax}} V(j,d)$, and the largest gain is calculated by $V(j, d_j^*)$.

During modeling, LightGBM can record the number of times each feature is used as a split node and the information gain after splitting. The feature importance can be quantified in this manner. This is a technique built into LightGBM and is often used to aid in feature selection. Refer to Ke et al. (2017) for more detailed explanations of LightGBM.

Artificial neural networks

- MLP

MLP is a simple ANN with a forward structure of the input layer, hidden layer, and output layer, as shown in Fig. 4(c). The layers are connected by weights, biases, and transfer functions. Except for the nodes in the input layer, each node has an activation function, which is used to introduce nonlinear factors, so that the neural network can approximate the nonlinear input–output mapping. MLPs are generally trained through backpropagation to assign and update the connection weights. Refer to Chianese et al. (2019) for more detailed explanations of MLP.

- DNN

When it comes to DNN, it is usually a very broad concept; CNNs, RNNs, etc. can all be classified as DNNs. But in this paper, DNN refers specifically to ordinary neural networks with multiple hidden layers. The MLP introduced above can be understood as one simplest DNN with only one hidden layer. By stacking multiple layers of non-linear transformations, DNNs can extract hierarchical representations of the time series data (Sezer et al. 2020). The increase in the number of hidden layers generally improves prediction accuracy, but the issues of over-fitting and gradient vanishing may also come along. In the later study of this paper, the effect of network "depth" can also be investigated by observing the prediction performance between MLP and DNN.

- CNN

CNN is mostly applied in image recognition (Lee et al. 2020) and natural language processing (Zhao et al. 2018). Some studies also used it for time series forecasting with its advantages in local feature mining (Wang et al. 2022). A basic CNN consists of five parts: input layer, convolutional layer, pooling layer, fully connected layer, and output layer, as shown in Fig. 4(e). Convolutional layers are used to extract features and reduce noise. Pooling layers are used to compress the amount of data and parameters to prevent over-fitting. Key techniques including local receptive field, weight sharing, and pooling are introduced, to simplify the networks and enhance the stability of the network against displacement, scaling, and nonlinear deformation. Refer to Kattenborn et al. (2021) for more detailed explanations of CNN.

- LSTM

LSTM is a highly representative Recurrent Neural Network (RNN). Its recurrent nature makes it particularly effective in modeling series data with long-term dependencies. Studies have shown that LSTM has achieved promising results in various time series prediction tasks, such as stock price prediction (Cao et al. 2019) and traffic flow prediction (Yang et al. 2019). LSTM has the memory to store knowledge through the joint action of a set of gates, namely forget gate, input gate, and output gate. Figure 4(d) shows a simplified framework of the LSTM. The forget gate discards the knowledge input from the previous cell. The remaining knowledge is further processed by the input gate, and the useful knowledge is input into the current cell. The output gate determines the final output of the current cell. Refer to Hochreiter and Schmidhuber (1997) for more detailed explanations of LSTM.

It is necessary to add that both LSTM and CNN have special requirements for the format of input data, which must be a 3D tensor with shape (samples, timesteps, features), while the input format of other models used in this study is a 2D (samples, features). Therefore, we need to rearrange the input data when dealing with LSTM and CNN.

In summary, the selected base predictors offer a diverse set of strengths that make them well-suited for our prediction task. XGBoost and LightGBM are tree models with powerful feature selection and missing data handling capabilities, while MLP, DNN, CNN, and LSTM are ANNs that excel at capturing nonlinear relationships and extracting high-level features from raw data. By leveraging the strengths of each model, they complement each other and improve the prediction accuracy and stability of the ensemble model.

Module 3: Dynamic multi-predictor ensemble

Dynamic model matching based on a comprehensive evaluation metric This study uses evaluation metrics to find the feature set that performs best for specific models. In the field of machine learning, Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Pearson Correlation Coefficient (PCC) are three mainstream metrics for model performance evaluation. MAE and RMSE have dimensions and are related to the magnitude of the data. PCC is dimensionless. They are calculated in Eqs. (6)–(8):

$$MAE = \left(\sum_{t=1}^{N} |Y_t - P_t| \right) / N \tag{6}$$

$$RMSE = \sqrt{\left(\sum_{t=1}^{N} (Y_t - P_t)^2 \right) / N} \tag{7}$$

$$PCC = \frac{\sum_{t=1}^{N} (Y_t - E_t(Y_t))(P_t - E_t(P_t))}{\sqrt{\sum_{t=1}^{N} (Y_t - E_t(Y_t))^2} \times \sqrt{\sum_{t=1}^{N} (P_t - E_t(P_t))^2}} \tag{8}$$

where Y_t is the actual observations in time t, and P_t is the predicted result in time t. N denotes the total number of observations. $E_t(Y_t)$ and $E_t(P_t)$ respectively denote the mean values of actual and prediction results.

However, single-metric evaluation suffers from the following limitations: (a) These three metrics are of different criteria. For instance, RMSE measures discreteness, while PCC measures linear correlation. In practice, however, it is often not the case that a model is optimal in all evaluation criteria. (b) These three metrics are of different scales. For instance, MAE and RMSE take values from 0 to $+\infty$. The smaller the value, the better the performance. PCC takes values from -1 to 1. The closer the absolute value is to 1, that is, the larger the absolute value, the better the performance.

To this end, this study proposes a comprehensive metric called $rank_{all}$. An example of the process of calculating this metric is presented in Fig. 5. First, consider the ranking of the metric of one feature set among all ten feature sets. The better the performance (in the customized standards of each metric), the higher the ranking. That is, MAEs and RMSEs need to be sorted in ascending order, while PCCs need to be sorted in descending order. Second, aggregate the rankings of these three metrics. It is important to note that in our study, we consider the metrics to be equally important, and therefore their rankings are weighted on an average basis. However, the weighting of the rankings can be flexibly adjusted to meet specific needs in other scenarios. For instance, if one prioritizes correlation, a higher weight can be assigned to PCC.

The calculation formula of $rank_{all}$ is given in Eq. (9):

$$rank^i_{all} = (rank^i_{mae} + rank^i_{rmse} + rank^i_{pcc})/3 \tag{9}$$

where the superscript i is for model i. The subscripts m, r, and p are for metrics MAE, RMSE, and PCC, respectively. $rank^i_{mae}$ denotes the ranking of MAE of model i among that of all the models in ascending order. $rank^i_{rmse}$ denotes the ranking of RMSE of model i among that of all the models in ascending order. denotes the ranking of PCC of model i among that of all the models in descending order.$rank^i_{mae}$, $rank^i_{rmse}$, and $rank^i_{pcc}$ all take integer values from 0 to 9.

Optimized ensemble based on OPTUNA OPTUNA is initially proposed to overcome the drawbacks of Grid Search (GS) and Random Search (RS), which are two conventional hyperparameter optimization methods. GS configures hyperparameters to be optimized to form a "grid" space and implements an exhaustive search in this space. RS proved to be less computationally costly than GS, as it explores in specified configurations rather than all the possible configurations (Panichella 2021). One major drawback of these two is that the search is aimless, and they do not determine whether the current domain is worth exploring, which means that a great deal of time may be spent in evaluating "bad" hyperparameter configurations (Pravin et al. 2022).

OPTUNA casts HPO as a process of minimizing or maximizing an objective function, in which the hyperparameters and their ranges are specified, and the validation score is returned (Akiba et al. 2019). One of the most powerful functions of OPTUNA is that it takes advantage of previous trials in the pre-defined search space to determine which trial to try next. If a trial is not very promising, it can be terminated early so that the search space is methodically narrowed, and more time is made for trials with better hyperparameters.

In the implementation of our study, the objective function is to minimize the RMSE evaluated on the validation set, and the hyperparameters to be optimized are the ensemble weights for each base predictor, with a search space of 0 to 1. It can generally be expressed by Eq. (10):

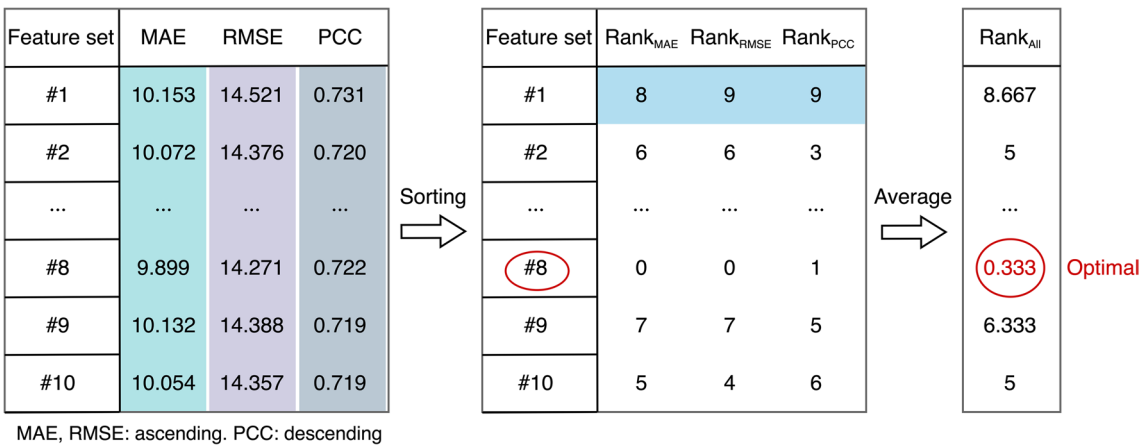

Fig. 5 Calculation process of $rank_{all}$

$$S^* = arg \min_{0\le\omega_1,\cdots\omega_6\le 1} \{RMSE[Y(t;D), P(t;D)]\}$$
$$s.t. P(t;D) = \frac{\sum_{a=1}^{6}(P_a(t;D)\times\omega_a)}{\sum_{a=1}^{6}\omega_a} \quad (10)$$

where S^* denotes the optimal hyperparameters that minimizes the objective function, i.e., a set of six weights. $Y(t;D)$ denotes the actual observation of sample D at time t.$P(t;D)$,$P_1(t;D)$,$P_2(t;D)$,$P_3(t;D)$, $P_4(t;D)$,$P_5(t;D)$, and $P_6(t;D)$ respectively denote the predicted values of the ensemble model, LightGBM, XGBoost, LSTM, CNN, MLP, and DNN for sample D at time t. $[\omega_1, \omega_2, \cdots, \omega_6]$ are the ensemble weights.

Case study

Dataset preparation

To validate the feasibility and robustness of the proposed method, experiments were carried out on four datasets from different Chinese cities. Table 2 gives an overview of the four datasets at the city level and a statistical description at the series level. The four selected cities are Suzhou, Nanchang, Shenzhen, and Urumqi, which are distinct in terms of geographical location and economic construction level. The statistical characteristics of the four series differ greatly. All above ensure the diversity of the experimental data.

Each series was collected at hourly resolution, spanning from 2020/01/01 to 2020/12/31, with a total of 8746 samples. In this study, missing values in the raw series are first filled by cubic interpolation as a simple preprocessing step. And then the series is divided into the training set, validation set, and testing set at the ratio of 3:1:1. The training set is used to train base predictors. The validation set is used to ascertain intermediate values, such as model configurations and ensemble weights. The testing set is used to ultimately evaluate the generalization ability of the model. Figure 6 shows the distribution of the AQI series for four selected cities, and the division of training, validation, and test sets.

Comparison analysis

Analysis of candidate features

A total of 46 candidate features are extracted. A summary of them is given in Table 3.

In this section, we adopt Pearson correlation analysis to explore how the candidate features as well as the target label are linearly correlated. Taking Dataset #1 as an example, the Pearson correlation between all pairs of variables (i.e., 46 candidate features and 1 label) is displayed in the form of a 47×47 matrix, as shown in Fig. 7, where the (i, j)th element is the PCC between the ith variable and the jth variable. The diagonal elements, representing the PCC between each variable and itself, are always 1. It can be seen from Fig. 7 that:

1) To analyze from the perspective of intra-class comparison: (a) A clear intra-class Pearson correlation is shown within the lag features (close to 1), which also confirms the autocorrelation of the time series mentioned above. (b) Among the window features, the intra-class Pearson correlation maintains a relatively high level. (c) Timestamp features have little intra-class correlation with each other. It can be easily explained by the fact that the timestamp features only provide knowledge at a calendar level, not an observations level.
2) To analyze from the perspective of inter-class comparison: Both differential features and timestamp features are weakly correlated with other class features. For the differential features, the possible reasons are as follows. On the one hand, the second-order differential feature used in this study only involves the observations of the prior two time steps. On the other hand, the essence of the difference is to make nonstationary time series stationary, so it implies an inevitable loss of information every time the difference is performed.
3) The last column of the correlation matrix shows the relationship between features and the label. To analyze from the perspective of how the features are related to the label: (a) The lag features are most relevant to the label, but as the number of lagged steps increases, the correlation gradually weakens. (b) The differential features, the kurtosis features, and the timestamp features have the lowest correlation with the label.

Table 2 Description of dataset

Dataset	City level			Series level					
	City	Location	Tier	Min	Max	Mean	Standard deviation	Skew	Kurt
#1	Suzhou	East	New first-tier	8.0	263.0	56.3985	33.7858	2.0518	5.9629
#2	Nanchang	Central	Second-tier	8.0	221.0	55.5668	29.6335	1.0023	1.6868
#3	Shenzhen	South	First-tier	8.0	132.0	33.3791	16.8074	0.9946	1.4496
#4	Urumuqi	Northwest	Third-tier	0.0	500.0	95.5516	90.3008	2.0608	3.9532

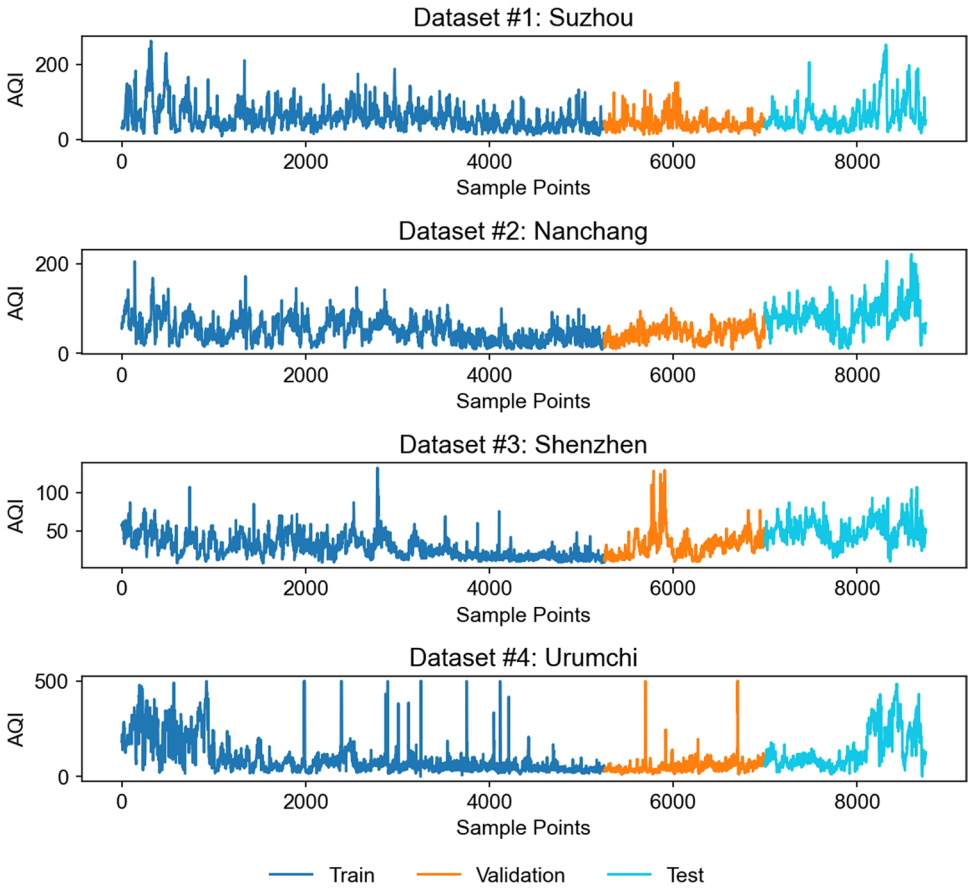

Fig. 6 Raw AQI series of four selected cities

Table 3 Summary of candidate features considered in this study

Type	Features	Brief description	Size
Timestamp features	*time_hour*	Hour of day, from 0 to 23, denoting 00:00 to 23:00	5
	time_daypart	Daypart of day, from 0 to 5, denoting dawn, morning, noon, afternoon, evening, and midnight, respectively	
	time_day	Day of week, from 0 to 6, denoting Monday to Sunday	
	time_day_of_month	Day of month	
	time_is_weekend	Whether it is a weekend	
Lag features	*lag t*	Observation at the t timestep before the current time step, where $t = 1, 2, 3, \ldots, 9$	9
Differential features	*diff1*	First-order difference of the observations at the current time step	2
	diff2	Second-order difference of the observations at the current time step	
Window features	*window_mean_S*	The mean of the observations over the previous window of size S, where $S = 12H, 2D, 3D, 5D, 10D$	30
	window_std_S	The standard deviation of the observations over the previous window of size S, where $S = 12H, 2D, 3D, 5D, 10D$	
	window_sum_S	The sum of the observations over the previous window of size S, where $S = 12H, 2D, 3D, 5D, 10D$	
	window_skew_ S	The skewness of the observations over the previous window of size S, where $S = 12H, 2D, 3D, 5D, 10D$	
	window_kurt_ S	The kurtosis of the observations over the previous window of size S, where $S = 12H, 2D, 3D, 5D, 10D$	
	window_max_S	The maximum of the observations over the previous window of size S, where $S = 12H, 2D, 3D, 5D, 10D$	

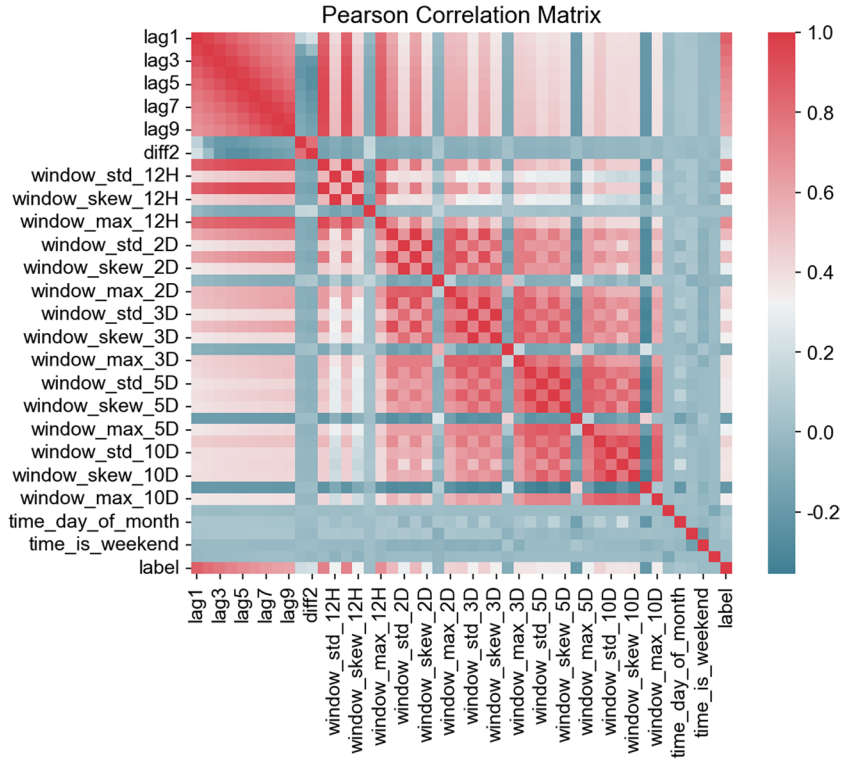

Fig. 7 Pearson correlation matric of all variables (including candidate features and label) of Dataset #1. Note that: Some tick labels have been omitted for brevity. The actual sequence of the matrix from top to bottom is listed in: *lag1*, *lag2*, *lag3*, *lag4*, *lag5*, *lag6*, *lag7*, *lag8*, *lag9*, *diff1*, *diff2*, *window_mean_12H*, *window_std_12H*, *window_sum_12H*, *window_skew_12H*, *window_kurt_12H*, *window_max_12H*, *window_mean_12H*, *window_std_12H*, *window_sum_12H*, *window_skew_12H*, *window_kurt_12H*, *window_max_12H*, *window_mean_2D*, *window_std_2D*, *window_sum_2D*, *window_skew_2D*, *window_kurt_2D*, *window_max_2D*, *window_mean_3D*, *window_std_3D*, *window_sum_3D*, *window_skew_3D*, *window_kurt_3D*, *window_max_3D*, *window_mean_5D*, *window_std_5D*, *window_sum_5D*, *window_skew_5D*, *window_kurt_5D*, *window_max_5D*, *window_mean_10D*, *window_std_10D*, *window_sum_10D*, *window_skew_10D*, *window_kurt_10D*, *window_max_10D*, label

Impact of feature merging strategies

In this section, the prediction performances of six base predictors with five feature merging strategies are compared. Figure 8 and Table S1 give the evaluation of the experimental results on the validation set using base metrics, namely MAE, RMSE, and PCC. Given the limited length of this paper, only the case of Dataset #1 is shown, and the other three are not provided here. From Fig. 8 and Table S1, it can be found that:

1) None of the feature sets dominates the other when using different individual evaluation metrics. For instance, for LightGBM, when using MAE, it is Feature set #4 giving the minimum value of 9.9752. When using RMSE, it is Feature set #5. This is consistent with the reason why the comprehensive evaluation metric $rank_{all}$ is proposed.
2) To verify the effectiveness of the feature merging strategies, we take Feature set #1, which only contains lag features, as a baseline and compare it with the other four feature sets. In most cases, the optimal feature sets are among Feature set #2, #3, #4, and #5, i.e., the ones that adopt feature merging strategies. It implies that merging features from different classes does have a positive contribution to predictive performance, outperforming features from a single class. On the other hand, model performance does not strictly improve with the continuous addition of features, but rather changes from good to bad. It implies that blindly adding features will bring some redundancy and complicate the model's understanding. The feature filtering is then recognized as a possible better solution, which will be presented and discussed in the "Impact of feature filtering strategies" section.
3) To analyze from the perspective of features, for most models (XGBoost, LSTM, CNN, MLP, DNN), the optimal feature set is either Feature set #1 or Feature set #2. One explanation for this can be that lag features and differential features transfer more valuable knowledge for model prediction. As observed above, lag features and differential features are less correlated. Therefore, when the two are merged, they can complement each other well so that the model can achieve higher accuracy.
4) To analyze from the perspective of models: (a) Opposite to ANNs and XGBoost, LightGBM can instead benefit from

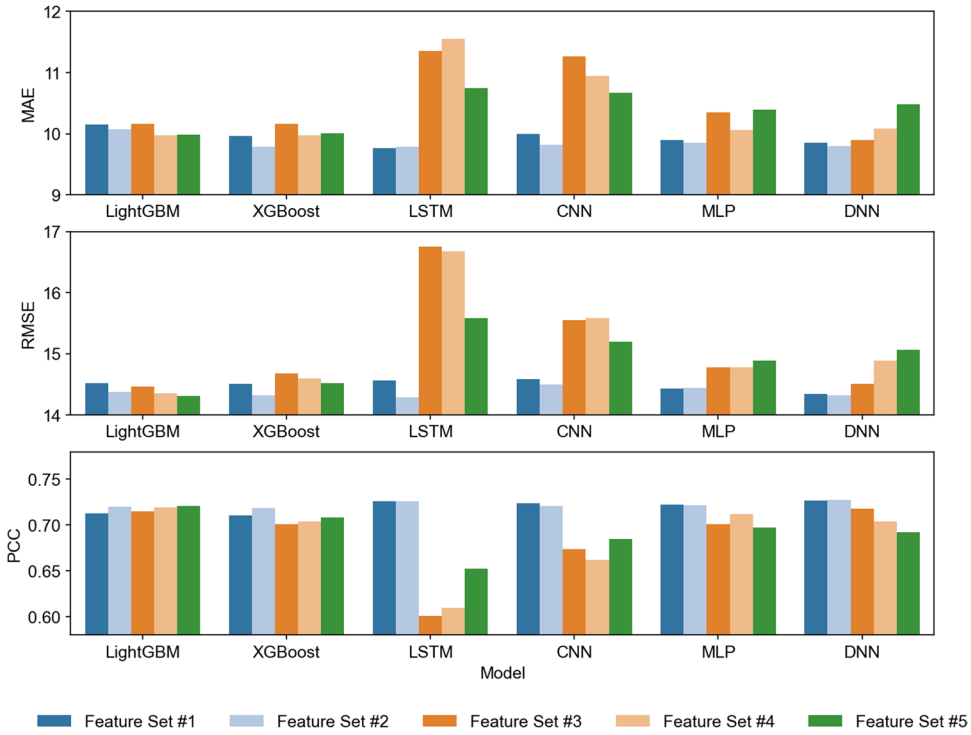

Fig. 8 Prediction evaluation metrics of base predictors with Feature sets #1, #2, #3, #4, and #5 (Dataset #1, validation set)

datasets with a larger number of feature variables. When performing on Feature set #5, it gives the minimum RMSE of 14.3124 and the maximum PCC of 0.7210. It is because LightGBM has an exceptional technique called Exclusive Feature Bundling (EFB); it treats features with mutually exclusive relationships as one to adapt to high-dimensional features and avoid redundancy (Alabdullah et al. 2022). (b) Among all ANNs, under all evaluation criteria, the performance of LSTM with its optimal feature set is better than that of other models. Moreover, DNN outperforms MLP, which proves that in general, the fitting capacity of ANNs strengthens with the increase in the number of hidden layers. Nevertheless, in most cases, ANNs perform inferior to tree models and are more sensitive to feature changes. This may be caused by the model structure or hyperparameter tuning that does not meet expectations.

Impact of feature filtering strategies

After training LightGBM with Feature set #5, a list of feature importance rankings for all candidates is output. For different datasets, the ranking results can be quite different. Taking Dataset #1 as an example, Fig. 9(a) shows the top 25 features

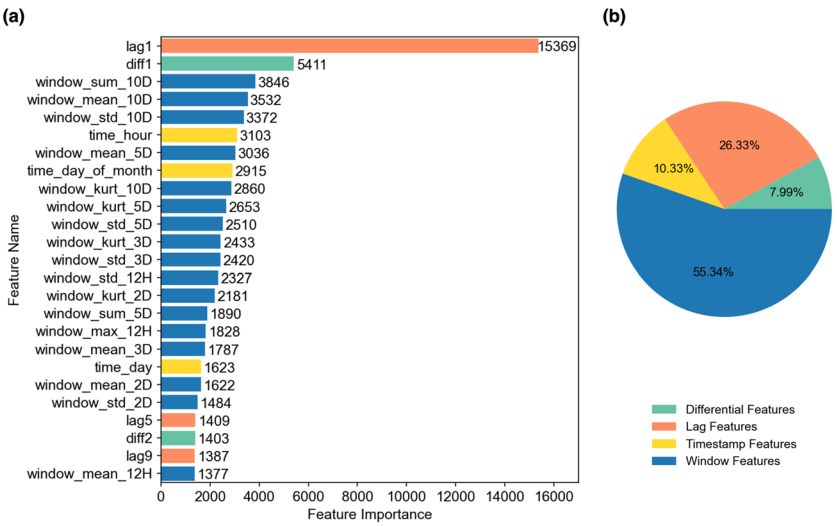

Fig. 9 Importance of candidate features: (**a**) The top 25 in descending order. (**b**) The importance ratios of four feature classes (Dataset #1, validation set)

in descending order of feature importance. Bars for features from different classes are distinguished by different colors. Figure 9(b) shows the ratios of feature importance of four feature classes. Lag features, differential features, window features, and timestamp features account for 26.3%, 8.0%, 55.3%, and 10.3% respectively. Table S2 shows the full sorting list of all features. From Fig. 9 and Table S2, it can be seen that:

1) The "*lag1*" is the most important feature, with importance of 15,369, far ahead of other features. The remaining lag features in the top 25 are "*lag5*" and "*lag9*," ranking 22nd and 24th respectively.
2) Among the differential features, the importance of the first-order differential feature "*diff1*" ranks 2nd, at 5411. The importance of the second-order differential feature "*diff2*" ranks 23rd.
3) The overall importance of window features is the highest, most of which are ranked between the top 3 and the top 21. The "*window_sum_10D*," "*window_mean_10D*," and "*window_std_10D*" are the most important, indicating that coarse-grained windows can better capture factors that are valuable for prediction. All kurtosis features, regardless of their window sizes, have an importance of 0.
4) The two most important timestamp features are "*time_hour*" (ranked 6th) and "*time_day*" (i.e., weekday, ranked 18th). Both are cyclical. And the importance of "*time_is_weekend*" is only 130, which shows the AQI has no obvious calendar effect.

To sum up, by looking at the feature importance rankings generated by the trained LightGBM, the contributions of individual features are paid attention to. There are weak features that are of low importance in each feature class. Therefore, if we merge features in classes, there will certainly be cases where both strong and weak features are fed into the model, resulting in redundancy and no significant improvement in the prediction performance of the model. Thus, the way of filtering out the contributing features to build feature sets is more reasonable and flexible than the way of simple inter-class combinations.

Figure 10 and Table S3 give the performance evaluation among the base predictors using different feature filtering strategies. It can be seen that:

1) To verify the effectiveness of the feature filtering strategies, we take Feature set #5, which contains all candidate features, as a baseline and compare it with the other five feature sets. In Fig. 10, the baseline is displayed in the form of a black dotted line. A clear improvement can be seen in all ANNs after feature filtering. The MAEs and RMSEs are smaller, and the PCCs are larger. But it is not the case with

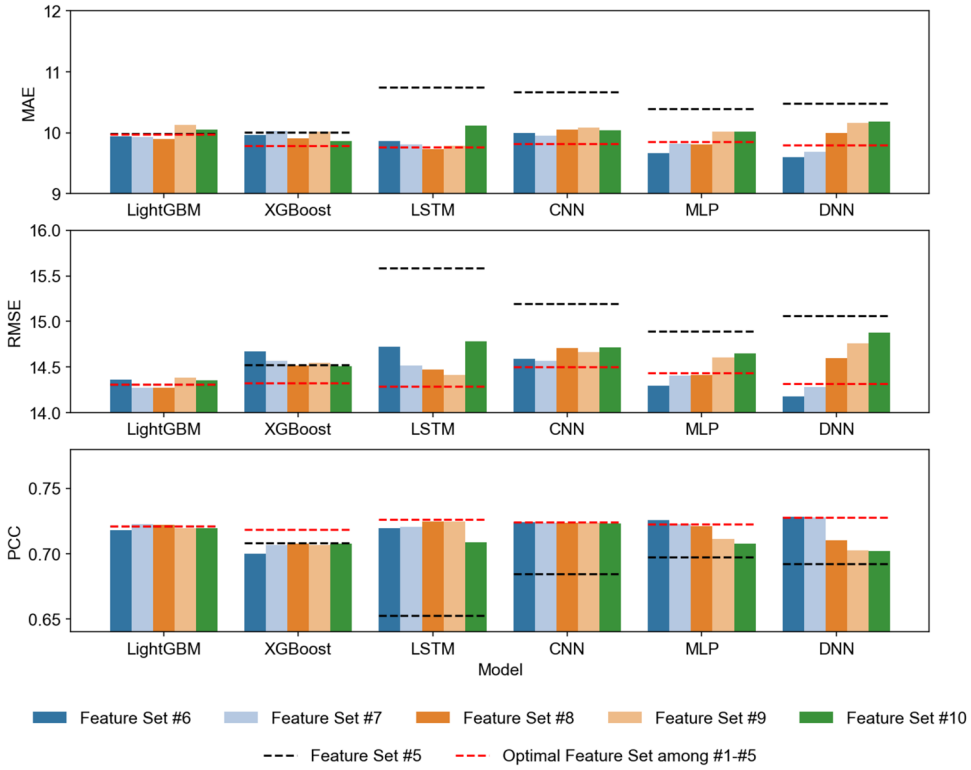

Fig. 10 Prediction evaluation metrics of base predictors with Feature sets #6, #7, #8, #9, and #10 (Dataset #1, validation set)

tree models. For instance, LightGBM performs worse on Feature set #9 (MAE is 10.1318) and Feature set #10 (MAE is 10.0542) than on Feature set #5 (MAE is 9.9888). The possible reason is that tree models require intensive feature input and can self-identify redundant features. The feature filtering method may seem unnecessary here.

2) To compare the effect between the feature filtering strategies and the feature merging strategies, we add the performance of base predictors with their optimal feature sets obtained from merging strategies, which is displayed in the form of a red dashed line. Some of the optimal feature sets are derived from the feature merging strategies. For instance, the optimal feature set of XGBoost is Feature set #2, which is performed better than Feature set #6–#10, while feature filtering strategies also produce effective feature sets. For instance, DNN performs better on Feature set #6 (MAE is 9.5981) and Feature set #7 (MAE is 9.6847) than Feature set #2 (MAE is 9.7990). Therefore, the results confirm that the feature filtering strategies do not always outperform the feature merging strategies.

Dynamic model matching

In this section, we use the comprehensive evaluation method to calculate the scores of all groups of base predictors and feature sets, which are given in Table 4. Those with the smallest $rank_{all}$ are regarded as optimal. In this way, the six base predictors matched with optimal features are obtained and subsequently used for the ensemble. We can conclude that the proposed feature engineering is effective overall. Eleven out of 24 optimal feature sets are the ones adopted filtering strategies and 7 are Feature set #2 (adopted merging strategies).

The newly defined $rank_{all}$ solves the deficiencies of the single-metric evaluation method in terms of criteria and scales. On the one hand, MAE, RMSE, and PCC are taken as base metrics. The contributions of each base metric are integrated in the way of a weighted average.

Table 4 Evaluation of all cases using the proposed metric $rank_{all}$

Dataset	Base predictor	Feature set									
		#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
#1	LightGBM	8.6667	5.0	8.3333	3.3333	2.6667	4.6667	0.6667	**0.3333**	6.3333	5
	XGBoost	1.6667	**0.0**	8.3333	6.3333	4.0	7.3333	6.3333	3.0	6.0	2
	LSTM	1.6667	**1.3333**	8.6667	8.3333	7.0	5.0	3.6667	1.6667	1.6667	6
	CNN	**1.6667**	2.0	8.3333	8.6667	7.0	2.0	2.0	4.3333	4.0	5
	MLP	2.6667	3.3333	7.6667	6.6667	9.0	**0.0**	1.6667	2.3333	5.3333	6.3333
	DNN	3.0	2.0	4.0	6.6667	9.0	**0.0**	1.0	5.0	6.6667	7.6667
#2	LightGBM	7.0	5.0	9.0	8.0	3.6667	**0.0**	1.0	4.0	4.6667	2.6667
	XGBoost	1.6667	3.6667	9.0	7.6667	7.0	**0.0**	2.6667	2.0	5.0	6.3333
	LSTM	2.6667	4.6667	8.3333	7.0	8.6667	3.3333	**0.3333**	2.6667	1.6667	5.6667
	CNN	3.6667	**0.3333**	7.0	8.0	9.0	3.3333	6.0	1.6667	4.0	2
	MLP	1.6667	**0.3333**	6.3333	7.0	9.0	3.0	1.0	4.0	6.0	6.6667
	DNN	1.3333	**0.0**	6.0	8.3333	8.6667	2.0	3.6667	3.0	6.3333	5.6667
#3	LightGBM	**0.3333**	1.0	7.3333	6.0	4.0	4.3333	9.0	7.0	2.6667	3.3333
	XGBoost	**0.3333**	0.6667	8.0	6.0	5.0	2.6667	5.3333	8.0	6.6667	2.3333
	LSTM	6.0	3.3333	9.0	7.3333	7.6667	4.6667	1.6667	2.3333	**1.3333**	1.6667
	CNN	4.3333	4.3333	9.0	7.0	8.0	3.3333	4.0	2.6667	**0.0**	2.3333
	MLP	**0.0**	1.0	5.0	6.6667	6.3333	2.0	3.0	4.3333	7.6667	9
	DNN	**0.3333**	0.6667	8.0	6.0	3.3333	2.0	3.6667	6.6667	9.0	5.3333
#4	LightGBM	7.0	8.6667	6.3333	6.0	3.3333	5.6667	**0.3333**	0.6667	3.0	4
	XGBoost	**1.6667**	2.6667	4.6667	7.3333	4.6667	3.3333	5.6667	9.0	4.3333	1.6667
	LSTM	4.0	**0.3333**	8.3333	8.6667	6.3333	6.0	1.3333	4.6667	1.3333	4
	CNN	5.0	**1.6667**	6.3333	4.3333	2.3333	5.3333	6.6667	2.0	4.6667	6.6667
	MLP	8.0	7.6667	6.6667	3.6667	5.0	3.3333	5.6667	**0.3333**	2.3333	2.3333
	DNN	6.3333	7.6667	1.6667	4.0	5.3333	4.3333	4.0	6.6667	**0.3333**	4.6667

The optimal feature sets for each predictor are shown in bold text

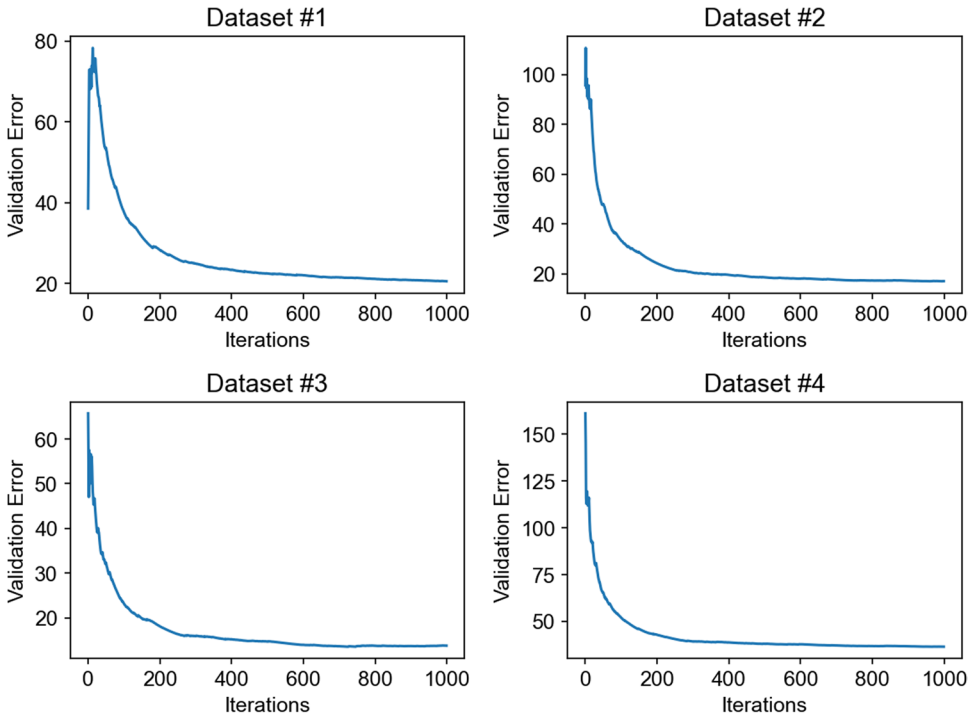

Fig. 11 Convergence curves of OPTUNA during iterations

The performance of multiple criteria is comprehensively evaluated. On the other hand, it normalizes the errors with large differences to the rankings with designated ranges. The impact of dimensions is eliminated as well. Also, this metric delivers better interpretability and intuition. Its value ranges from 0 to 1. A smaller value indicates a better performance.

Proposed model vs. single models

Figure 11 shows the convergence curves of OPTUNA during the process of optimizing the ensemble weights on the validation sets. Figure 12 and Table 5 compare the model performance on the testing sets under the cases where the six base predictors are used alone vs. the six are ensembled by

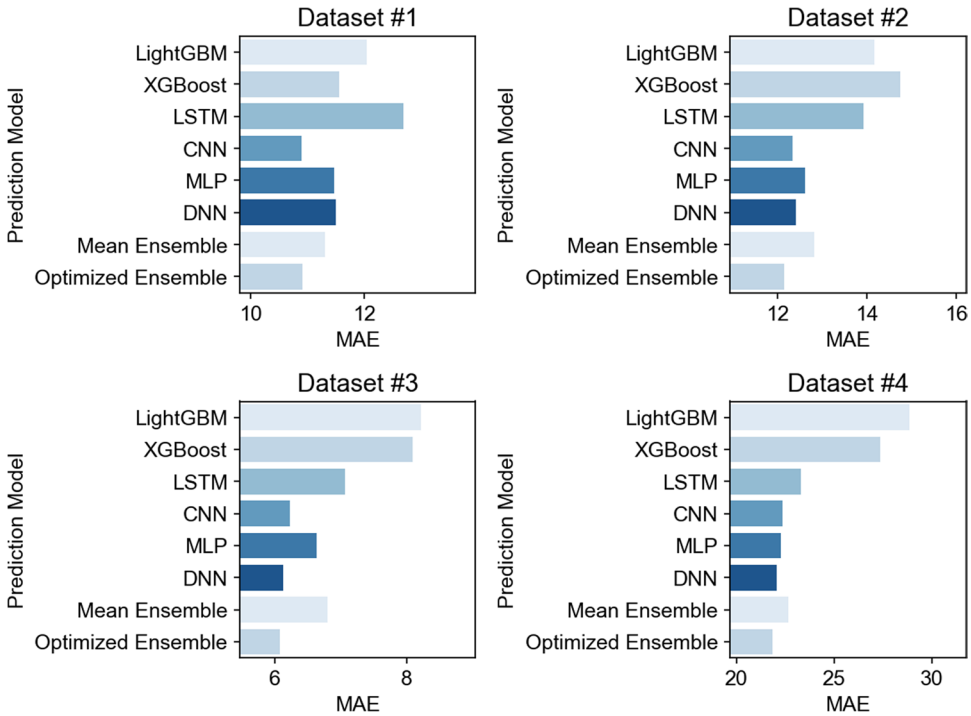

Fig. 12 Evaluation of eight base models, mean ensemble model, and optimized ensemble model on the testing sets (MAE)

Table 5 Evaluation of eight base models, mean ensemble model, and optimized ensemble model on the testing sets

Dataset	Base predictor	Optimal feature set	MAE	RMSE	PCC
#1	LightGBM	#8	12.0409	17.6659	0.9141
	XGBoost	#2	11.5534	17.0431	0.9152
	LSTM	#2	12.6915	18.5476	0.9205
	CNN	#1	10.8984	16.1695	0.9215
	MLP	#6	11.4763	16.8926	0.9155
	DNN	#6	11.5004	16.9261	0.9172
	Mean ensemble	/	11.3047	16.5697	0.9227
	Optimized ensemble	**/**	**10.9042**	**15.9836**	**0.9234**
#2	LightGBM	#6	14.1694	20.0062	0.8156
	XGBoost	#6	14.7383	21.3497	0.7908
	LSTM	#9	13.9294	19.6808	0.8307
	CNN	#2	12.3355	17.1566	0.8563
	MLP	#2	12.6125	17.6508	0.8508
	DNN	#2	12.4026	17.3142	0.8545
	Mean ensemble	/	12.8124	17.9425	0.8525
	Optimized ensemble	**/**	**12.1511**	**16.8598**	**0.8572**
#3	LightGBM	#1	8.2236	10.9524	0.7182
	XGBoost	#1	8.086	10.836	0.7306
	LSTM	#9	7.0688	9.5078	0.8087
	CNN	#9	6.2375	8.6454	0.8103
	MLP	#1	6.6329	9.0668	0.8081
	DNN	#1	6.1324	8.5711	0.8078
	Mean ensemble	/	6.8038	9.192	0.8011
	Optimized ensemble	**/**	**6.0816**	**8.4187**	**0.8103**
#4	LightGBM	#7	28.8691	44.7675	0.9125
	XGBoost	#1	27.3787	41.8499	0.9295
	LSTM	#2	23.3262	35.9732	0.9404
	CNN	#2	22.3807	36.4739	0.9376
	MLP	#8	22.2753	35.5991	0.9401
	DNN	#9	22.057	34.6938	0.9431
	Mean ensemble	/	22.6634	35.4941	0.9425
	Optimized ensemble	**/**	**21.8749**	**34.5657**	**0.9432**

The best performance among all models is shown in bold text

averaging vs. the six are ensembled by OPTUNA algorithm. Table 6 gives the ensemble weights of each base predictor generated by OPTUNA. Figure 13 compares the predicted results with the actual observations on the testing sets. We can conclude that:

1) The error on the validation set gradually decreases with the iterations of OPTUNA going on. This indicates that the optimization works and has a good robustness.
2) In all cases, the optimized ensemble model is superior to single models and the mean ensemble model. And the optimized ensemble model is more robust to different datasets.
3) The mean ensemble does not necessarily guarantee a good performance, and it may not be as effective as single models, such as CNN. It is because the mean ensemble cannot adaptively assign different weights to the base predictors based on the data. In contrast, the optimized ensemble can effectively cope with datasets of different distributions. As shown in Table 6, the weights for CNN are higher in Dataset #1, #2, and #3, which are 0.4987, 0.4807, and 0.4888 respectively. However, in Dataset #4, CNN is only given a weight of 0.0017 because of its poor performance.
4) The geographical location and industrial factors of Urumqi City result in Dataset #4 having the highest AQI magnitude and variance among all datasets, leading to the maximum MAE (21.8749) and RMSE (34.5657) produced by the optimized ensemble model. However, despite this challenging dataset, the proposed method demonstrates exceptional performance in terms of PCC (0.9432), showcasing its superiority in dealing with highly non-stationary data. For datasets with inferior prediction results, such as Dataset #3, further hyperparameter tuning may be necessary, as its optimized model only achieves a PCC of 0.8103.
5) From the prediction results of the proposed model shown in Fig. 13, the difference between the predicted value and the actual observations is very small. And it can make more accurate tracking for some mutation points and peak values.

Overall, the results highlight the promising potential of the proposed method for accurately predicting AQI across various cities and datasets.

Conclusions

In this study, an hourly, 3-step-ahead deterministic AQI prediction model involving feature engineering, dynamic model matching, automatic optimized ensemble is proposed and evaluated on datasets acquired from four different cities in China. We have favorable findings as follows:

1) The "extract-merge-filter" procedure provides valuable information to the model and each stage of it is proved to be indispensable in improving the overall performance. Potential information in the time series is fully exploited, and some groups of features (especially lag and differential features) are shown to be complementary, making the models stronger nonlinear approximators. The feature filtering plays the most important role, which is embodied in the fact that the optimal feature sets of 11 out of 24 cases are based on the filtering strategies.

Table 6 Ensemble weights of each base predictor

Dataset	LightGBM	XGBoost	LSTM	CNN	MLP	DNN
#1	0.0646	0.2354	0.2504	0.4987	0.0012	0.0016
#2	0.0388	0.1106	0.0205	0.4807	0.036	0.3614
#3	0.0012	0.036	0.4833	0.4888	0.0598	0.0021
#4	0.1053	0.1809	0.2175	0.0017	0.2264	0.3017

2) It is more reasonable to input customized features for different base predictors than to input the same ones. The comprehensive metric that helps to achieve this integrates the evaluation results of three metrics and is considered more reliable than when the three metrics are used separately. In addition to architecture optimization within the model, OPTUNA is also well suited to optimize the ensemble weights between multi-predictors. The optimized ensemble outperformed the single machine learning models and surpassed their averaged results.
3) The proposed model performed well on four distinct datasets, demonstrating the feasibility and robustness of the model. More importantly, due to the sufficient works on AQI features, influencing factors can be ascertained from the prediction results and further help to implement some practical pollution prevention and control measures.

There is, of course, scope for future studies to put effort on: (a) Previous studies have demonstrated the effectiveness of meteorological factors, such as temperature and humidity, as features in predicting AQI. Therefore, incorporating these variables into the proposed model and investigating their contributions to AQI prediction could be an area for further research. (b) To further enhance model accuracy, additional groups of features could be considered during the feature merging and filtering stages. For instance, using a finer granularity of feature filtering, with a change to a level of 1, could yield more reliable results and prevent the model from ending up in a local optimum. (c) The hyperparameter tuning within the models was not investigated, which may limit the model's performance on certain datasets and models. More advanced hyperparameter tuning methods remain to be explored.

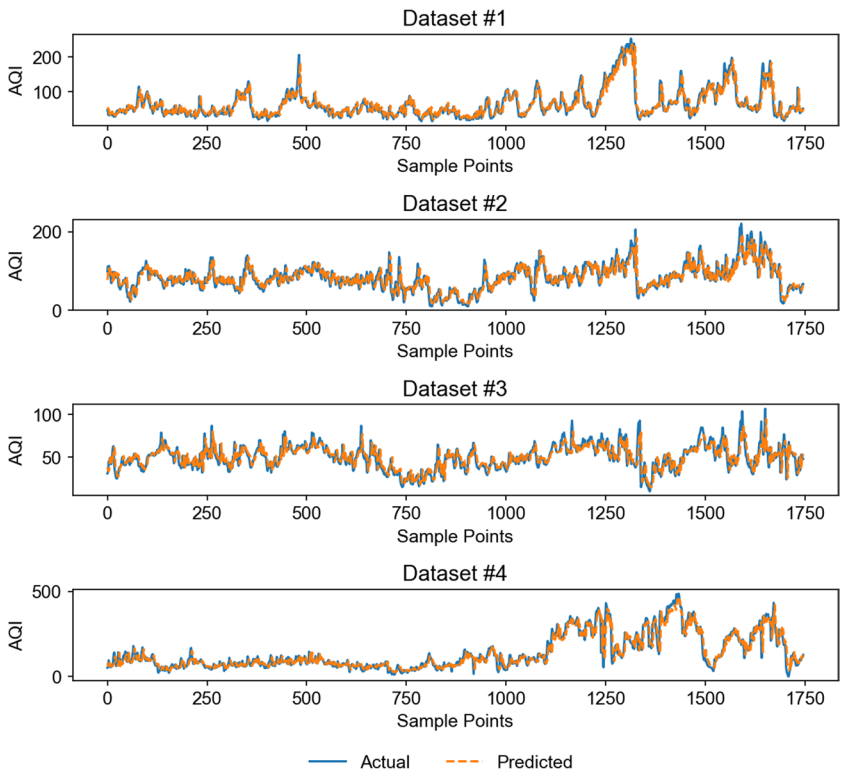

Fig. 13 Comparison of the predicted results with actual observations on the testing sets

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11869-023-01380-7.

Funding The study is fully supported by the National Natural Science Foundation of China (Grant No. 52072412), the Changsha Science & Technology Project (Grant No. KQ1707017), and the Hunan Province Science and Technology Talent Support Project (Grant No. 2020TJ-Q06).

Data availability The data that support the findings of this study are available from the corresponding author on reasonable request.

Declarations

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent to publish Not applicable.

Competing interests The authors declare no competing interests.

References

Akiba T, Sano S, Yanase T, Ohta T and Koyama M (2019) Optuna: a next-generation hyperparameter optimization framework. In Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. ACM, Anchorage, AK, USA, pp. 2623–2631. https://doi.org/10.1145/3292500.3330701

Alabdullah AA, Iqbal M, Zahid M, Khan K, Amin MN, Jalal FE (2022) Prediction of rapid chloride penetration resistance of metakaolin based high strength concrete using light GBM and XGBoost models by incorporating shap analysis. Constr Build Mater 345:128296. https://doi.org/10.1016/j.conbuildmat.2022.128296

Bitencourt HV, Orang O, de Souza LAF, Silva PC, Guimarães FG (2022) An embedding-based non-stationary fuzzy time series method for multiple output high-dimensional multivariate time series forecasting in Iot applications. Neural Comput Applic 35:9407–9420. https://doi.org/10.1007/s00521-022-08120-5

Cao J, Li Z, Li J (2019) Financial time series forecasting model based on CEEMDAN and LSTM. Physica A 519:127–139. https://doi.org/10.1016/j.physa.2018.11.061

Castelli M, Clemente FM, Popovič A, Silva S, Vanneschi L (2020) A machine learning approach to predict air quality in California. Complexity 2020:8049504. https://doi.org/10.1155/2020/8049504

Chianese E, Camastra F, Ciaramella A, Landi TC, Staiano A, Riccio A (2019) Spatio-temporal learning in predicting ambient particulate matter concentration by multi-layer perceptron. Eco Inform 49:54–61. https://doi.org/10.1016/j.ecoinf.2018.12.001

Dai H, Huang G, Wang J, Zeng H, Zhou F (2021) Prediction of air pollutant concentration based on one-dimensional multi-scale CNN-LSTM considering spatial-temporal characteristics: a case study of Xi'an. China Atmosphere 12:1626. https://doi.org/10.3390/atmos12121626

de Gennaro G, Trizio L, Di Gilio A, Pey J, Pérez N, Cusack M, Alastuey A, Querol X (2013) Neural network model for the prediction of PM10 daily concentrations in two sites in the Western Mediterranean. Sci Total Environ 463:875–883. https://doi.org/10.1016/j.scitotenv.2013.06.093

Domashova J, Mikhailina N (2021) Usage of machine learning methods for early detection of money laundering schemes. Procedia Comput Sci 190:184–192. https://doi.org/10.1016/j.procs.2021.06.033

Elmaz F, Eyckerman R, Casteels W, Latré S, Hellinckx P (2021) CNN-LSTM architecture for predictive indoor temperature modeling. Build Environ 206:108327. https://doi.org/10.1016/j.buildenv.2021.108327

Eslami E, Salman AK, Choi Y, Sayeed A, Lops Y (2020) A data ensemble approach for real-time air quality forecasting using extremely randomized trees and deep neural networks. Neural Comput Appl 32:7563–7579. https://doi.org/10.1007/s00521-019-04287-6

Guan W-J, Zheng X-Y, Chung KF, Zhong N-S (2016) Impact of air pollution on the burden of chronic respiratory diseases in China: time for urgent action. Lancet 388:1939–1951. https://doi.org/10.1016/S0140-6736(16)31597-5

Guo C, Liu G, Chen C-H (2020) Air pollution concentration forecast method based on the deep ensemble neural network. Wirel Commun Mob Comput 2020:8854649. https://doi.org/10.1155/2020/8854649

Hao Y, Tian C (2019) The study and application of a novel hybrid system for air quality early-warning. Appl Soft Comput 74:729–746. https://doi.org/10.1016/j.asoc.2018.09.005

He B-J, Ding L, Prasad D (2019) Enhancing urban ventilation performance through the development of precinct ventilation zones: a case study based on the Greater Sydney, Australia. Sustain Cities Soci 47:101472. https://doi.org/10.1016/j.scs.2019.101472

He X, Zhao K, Chu X (2021) Automl: a survey of the state-of-the-art. Knowl Based Syst 212:106622. https://doi.org/10.1016/j.knosys.2020.106622

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Jamei M, Ali M, Malik A, Karbasi M, Sharma E, Yaseen ZM (2022) Air quality monitoring based on chemical and meteorological drivers: application of a novel data filtering-based hybridized deep learning model. J Clean Prod 374:134011. https://doi.org/10.1016/j.jclepro.2022.134011

Ji C, Zhang C, Hua L, Ma H, Nazir MS, Peng T (2022) A multi-scale evolutionary deep learning model based on Ceemdan, improved whale optimization algorithm, regularized extreme learning machine and LSTM for AQI prediction. Environ Res 215:114228. https://doi.org/10.1016/j.envres.2022.114228

Kattenborn T, Leitloff J, Schiefer F, Hinz S (2021) Review on convolutional neural networks (CNN) in vegetation remote sensing. ISPRS J Photogramm Remote Sens 173:24–49. https://doi.org/10.1016/j.isprsjprs.2020.12.010

Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q and Liu T-Y (2017) LightGBM: a highly efficient gradient boosting decision tree. In Proceedings of the 31st International Conference on Neural Information Processing Systems. Curran Associates Inc., Long Beach, California, USA, pp 3149–3157

Kim D, Han H, Wang W, Kang Y, Lee H, Kim HS (2022) Application of deep learning models and network method for comprehensive air-quality index prediction. Appl Sci 12:6699. https://doi.org/10.3390/app12136699

Lee S, Kim H, Lieu QX, Lee J (2020) CNN-based image recognition for topology optimization. Knowl Based Syst 198:105887. https://doi.org/10.1016/j.knosys.2020.105887

Li J, Hao J, Feng Q, Sun X, Liu M (2021) Optimal selection of heterogeneous ensemble strategies of time series forecasting with multi-objective programming. Expert Syst Appl 166:114091. https://doi.org/10.1016/j.eswa.2020.114091

Li R, Jin Y (2018) The early-warning system based on hybrid optimization algorithm and fuzzy synthetic evaluation model. Inf Sci 435:296–319. https://doi.org/10.1016/j.ins.2017.12.040

Li S, Xie G, Ren J, Guo L, Yang Y, Xu X (2020) Urban PM2. 5 concentration prediction via attention-based CNN–LSTM. Appl Sci 10:1953. https://doi.org/10.3390/app10061953

Li Y, Peng T, Hua L, Ji C, Ma H, Nazir MS, Zhang C (2022) Research and application of an evolutionary deep learning model based on improved grey wolf optimization algorithm and DBN-ELM for AQI prediction. Sustain Cities Soc 87:104209. https://doi.org/10.1016/j.scs.2022.104209

Liu C-M (2002) Effect of PM2. 5 on AQI in Taiwan. Environ Model Softw 17:29–37. https://doi.org/10.1016/S1364-8152(01)00050-0

Liu D-R, Hsu Y-K, Chen H-Y, Jau H-J (2021a) Air pollution prediction based on factory-aware attentional LSTM neural network. Computing 103:75–98. https://doi.org/10.1007/s00607-020-00849-y

Liu H, Chen C (2019) Data processing strategies in wind energy forecasting models and applications: a comprehensive review. Appl Energy 249:392–408. https://doi.org/10.1016/j.apenergy.2019.04.188

Liu H, Xu Y, Chen C (2019) Improved pollution forecasting hybrid algorithms based on the ensemble method. Appl Math Model 73:473–486. https://doi.org/10.1016/j.apm.2019.04.032

Liu H, Yan G, Duan Z, Chen C (2021b) Intelligent modeling strategies for forecasting air quality time series: a review. Appl Soft Comput 102:106957. https://doi.org/10.1016/j.asoc.2020.106957

Liu H, Yang R (2021) A spatial multi-resolution multi-objective data-driven ensemble model for multi-step air quality index forecasting based on real-time decomposition. Comput Ind 125:103387. https://doi.org/10.1016/j.compind.2020.103387

Liu X, Qin M, He Y, Mi X, Yu C (2021c) A new multi-data-driven spatiotemporal PM2. 5 forecasting model based on an ensemble graph reinforcement learning convolutional network. Atmos Pollut Res 12:101197. https://doi.org/10.1016/j.apr.2021.101197

Luo Z, Huang J, Hu K, Li X and Zhang P (2019) Accuair: winning solution to air quality prediction for KDD Cup 2018. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, Anchorage, AK, USA, pp 1842–1850. https://doi.org/10.1145/3292500.3330787

Masmoudi S, Elghazel H, Taieb D, Yazar O, Kallel A (2020) A machine-learning framework for predicting multiple air pollutants' concentrations via multi-target regression and feature selection. Sci Total Environ 715:136991. https://doi.org/10.1016/j.scitotenv.2020.136991

Ojagh S, Cauteruccio F, Terracina G, Liang SH (2021) Enhanced air quality prediction by edge-based spatiotemporal data preprocessing. Comput Electr Eng 96:107572. https://doi.org/10.1016/j.compeleceng.2021.107572

Panichella A (2021) A systematic comparison of search-based approaches for LDA hyperparameter tuning. Inf Softw Technol 130:106411. https://doi.org/10.1016/j.infsof.2020.106411

Perez P, Menares C (2018) Forecasting of hourly PM2. 5 in south-west zone in Santiago De Chile. Aerosol Air Qual Res 18:2666–2679. https://doi.org/10.4209/aaqr.2018.01.0029

Pravin P, Tan JZM, Yap KS, Wu Z (2022) Hyperparameter optimization strategies for machine learning-based stochastic energy efficient scheduling in cyber-physical production systems. Digital Chem Eng 4:100047. https://doi.org/10.1016/j.dche.2022.100047

Sezer OB, Gudelek MU, Ozbayoglu AM (2020) Financial time series forecasting with deep learning: a systematic literature review: 2005–2019. Appl Soft Comput 90:106181. https://doi.org/10.1016/j.asoc.2020.106181

Singh KP, Gupta S, Rai P (2013) Identifying pollution sources and predicting urban air quality using ensemble learning methods. Atmos Environ 80:426–437. https://doi.org/10.1016/j.atmosenv.2013.08.023

Sipper M, Moore JH (2022) AddGBoost: a gradient boosting-style algorithm based on strong learners. Mach Learn Appl 7:100243. https://doi.org/10.1016/j.mlwa.2021.100243

Song K, Yan F, Ding T, Gao L, Lu S (2020) A steel property optimization model based on the XGBoost algorithm and improved PSO. Comput Mater Sci 174:109472. https://doi.org/10.1016/j.commatsci.2019.109472

Srinivas P, Katarya R (2022) Hyoptxg: optuna hyper-parameter optimization framework for predicting cardiovascular disease using XGBoost. Biomed Signal Process Control 73:103456. https://doi.org/10.1016/j.bspc.2021.103456

Surakhi O, Zaidan MA, Fung PL, Hossein Motlagh N, Serhan S, AlKhanafseh M, Ghoniem RM, Hussein T (2021) Time-lag selection for time-series forecasting using neural network and heuristic algorithm. Electronics 10:2518. https://doi.org/10.3390/electronics10202518

Thongthammachart T, Araki S, Shimadera H, Matsuo T, Kondo A (2022) Incorporating light gradient boosting machine to land use regression model for estimating NO2 and PM2. 5 Levels in Kansai Region, Japan. Environ Model Softw 155:105447. https://doi.org/10.1016/j.envsoft.2022.105447

Wang J, Jin L, Li X, He S, Huang M, Wang H (2022) A hybrid air quality index prediction model based on CNN and attention gate unit. IEEE Access. https://doi.org/10.1109/ACCESS.2022.3217242

Wu C-f, Larson TV, Wu S-y, Williamson J, Westberg HH, Liu L-JS (2007) Source apportionment of PM2. 5 and selected hazardous air pollutants in Seattle. Sci Total Environ 386:42–52. https://doi.org/10.1016/j.scitotenv.2007.07.042

Wu L, Gao X, Xiao Y, Liu S, Yang Y (2017) Using grey Holt-Winters model to predict the air quality index for cities in China. Nat Hazards 88:1003–1012. https://doi.org/10.1007/s11069-017-2901-8

Xian S, Chen K, Cheng Y (2022) Improved seagull optimization algorithm of partition and XGBoost of prediction for fuzzy time series forecasting of COVID-19 daily confirmed. Adv Engin Softw 173:103212. https://doi.org/10.1016/j.advengsoft.2022.103212

Yang B, Sun S, Li J, Lin X, Tian Y (2019) Traffic flow prediction using LSTM with feature enhancement. Neurocomputing 332:320–327. https://doi.org/10.1016/j.neucom.2018.12.016

Yang Y, Zheng Z, Bian K, Song L, Han Z (2017) Real-time profiling of fine-grained air quality index distribution using UAV Sensing. IEEE Internet Things J 5:186–198. https://doi.org/10.1109/JIOT.2017.2777820

Zhang K, Thé J, Xie G, Yu H (2020) Multi-step ahead forecasting of regional air quality using spatial-temporal deep neural networks: a case study of Huaihai Economic Zone. J Clean Prod 277:123231. https://doi.org/10.1016/j.jclepro.2020.123231

Zhang L, Lin J, Qiu R, Hu X, Zhang H, Chen Q, Tan H, Lin D, Wang J (2018) Trend analysis and forecast of PM2. 5 in Fuzhou, China using the ARIMA model. Ecol Ind 95:702–710. https://doi.org/10.1016/j.ecolind.2018.08.032

Zhao S, Xu Z, Liu L, Guo M, Yun J (2018) Towards accurate deceptive opinions detection based on word order-preserving CNN. Math Probl Eng 2018:2410206. https://doi.org/10.1155/2018/2410206

Zhao X, Li Q, Xue W, Zhao Y, Zhao H, Guo S (2022) Research on ultra-short-term load forecasting based on real-time electricity price and window-based XGBoost model. Energies 15:7367. https://doi.org/10.3390/en15197367

Zhou G, Xu J, Xie Y, Chang L, Gao W, Gu Y, Zhou J (2017) Numerical air quality forecasting over eastern China: an operational application of WRF-Chem. Atmos Environ 153:94–108. https://doi.org/10.1016/j.atmosenv.2017.01.020

Zhu S, Yang L, Wang W, Liu X, Lu M, Shen X (2018) Optimal-combined model for air quality index forecasting: 5 cities in North China. Environ Pollut 243:842–850. https://doi.org/10.1016/j.envpol.2018.09.025