fire

MDPI

Article

Cable Fire Risk Prediction via Dynamic Q-Learning-Driven Ensemble of Deep Temporal Networks

Haoxuan Li [1], Hao Gao [1,2], Xuehong Gao [1] and Guozhong Huang [1,*]

[1] Research Institute of Macro-Safety Science, University of Science and Technology Beijing, Beijing 100083, China; joey_li@xs.ustb.edu.cn (H.L.); gaohao2336@boe.com.cn (H.G.); gaoxh2020@ustb.edu.cn (X.G.)
[2] Beijing BOE Chuang Yuan Technology Co., Ltd., Beijing 100176, China
* Correspondence: hjxhuanggz@ustb.edu.cn

Abstract

Cables, which are critical for power and signal transmission in complex buildings and underground infrastructure, are exposed to elevated fire risks during operation, making reliable risk prediction essential for building fire safety. This study proposes a multivariate cable fire risk prediction model that integrates three deep temporal networks (RNN, LSTM, and GRU) through a Q-learning-based ensemble learning (QBEL). The model uses current, voltage, power, temperature, humidity, oxygen concentration, and system risk values acquired from an intelligent fire alarm system as inputs. Using a real-world dataset comprising 3060 seven-dimensional time steps collected from a tobacco logistics center, QBEL achieves a test-set MSE of 1.73, RMSE of 1.31, MAE of 0.84, and MAPE of 2.66%, improving the MAE and MAPE of the best single recurrent network by approximately 10–12%. Comparative experiments against conventional ensemble approaches based on XGBoost (Python package, version 3.0.0) boosting and stacking, as well as recent time-series forecasting models including DLinear, PatchTST, MoLE, and Fredformer, demonstrate that QBEL attains the lowest MAE and MAPE among all methods, while maintaining an MSE close to that of the best linear baseline and a moderate computational cost of approximately 5.5×10^{-3} GFLOPs and 45 MB of memory per inference. These results indicate that QBEL provides a favorable balance between prediction accuracy and computational efficiency, supporting its potential use in edge-oriented monitoring pipelines for timely cable fire risk warnings in building environments.

Keywords: cable fire risk; time series prediction; ensemble learning; Q-learning

Check for updates

Academic Editor: Tiago Miguel Ferreira

Received: 27 November 2025
Revised: 6 January 2026
Accepted: 13 January 2026
Published: 29 January 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Cables, as pivotal infrastructure for power and signal transmission, are indispensable in modern complex buildings and underground infrastructure. Their extensive use spans diverse sectors, industries, and enterprises. Concurrently, issues related to the safe operation and reliability of cables have emerged as focal concerns, attracting widespread attention across society [1]. Given the large quantity and dense installation of power cables, together with the heat generated during operation, these components present a heightened fire risk. They are widely regarded as important ignition sources in buildings. Among all electrical products, cable-related incidents account for the highest proportion of fire occurrences [2], underscoring the critical role of cables in mitigating fire risk. Being inherently combustible, cables can propagate fire rapidly once ignited [3], posing severe threats to

human life and property. Therefore, forecasting and providing early warnings for cable fires is of paramount importance.

Internet of Things (IoT) technology has been widely applied in fire detection systems for complex buildings, utilizing temperature and smoke detectors; however, it suffers from delayed alarm responses [4]. Conventional fire detection methods primarily rely on temperature and smoke sensing [5]. Typically, alarms are triggered only when fire-induced physical and chemical changes in the surrounding environment reach predefined thresholds, which hinders the timely implementation of effective preventive measures. As a result, real-time monitoring of cable electrical operating states via multiple sensors has emerged as a preventive approach to cable fire hazards [6]. Nevertheless, real-time monitoring data mainly reflect the current fire risk and lack advanced predictive capability. Fire risk assessment and mitigation in buildings and urban areas, therefore, remain critical global challenges, necessitating dynamic and data-driven tools to support proactive safety management [7].

In recent years, artificial intelligence technologies have developed rapidly [8] and have increasingly been applied to fire hazard warning systems [9]. Machine learning, as a core branch of artificial intelligence, has achieved notable advances in time-series forecasting [10]. Such methods train models on historical time-series data to predict future system states. The early stages of cable fires involve a gradual process of risk evolution, which can be reflected by subtle, coupled changes in electrical operating parameters and cabinet microclimate before pronounced alarm signatures appear [11]. For multi-sensor time-series data describing cable operating conditions, time-series forecasting techniques can be employed to predict fire risk, thereby enabling proactive prevention. In operational monitoring, a short prediction horizon of 10 to 30 s can be meaningful for risk-informed actions and alarm confirmation, but insufficient predictive accuracy remains a common challenge.

The motivation of this study is to address the limitations of existing cable fire warning systems, including delayed alerts and suboptimal predictive performance. To this end, a Q-learning-based ensemble learning method is proposed for predicting cable fire risk. Unlike traditional ensemble approaches (e.g., weighted averaging or stacking), the proposed method dynamically adjusts model weights through real-time interaction with temporal prediction errors. This mechanism enables adaptive compensation for the limitations of individual base learners (RNN, LSTM, and GRU) under varying risk evolution patterns, a capability absent in static ensemble frameworks. The main contributions of this study are summarized as follows. First, in response to the delayed alerts of traditional fire detection methods, this work focuses on the proactive prediction of electrical fire risks based on the operating states of cables. Second, a multivariate model is introduced to capture interactions among multiple risk-related attributes, making it well-suited for intelligent fire alarm systems equipped with multi-sensor configurations. The model incorporates current, voltage, power, ambient temperature, humidity, oxygen content, and system risk values, forming a multivariate time series derived from heterogeneous sensor data. Third, to enhance predictive accuracy, a Q-learning-driven ensemble strategy is developed to effectively combine the outputs of three deep temporal networks, thereby improving overall risk prediction performance.

The remainder of this paper is organized as follows. Section 2 reviews related work on cable fire detection, time-series forecasting, and hybrid modeling approaches. Section 3 describes the proposed QBEL framework, including the base learners (RNN, LSTM, and GRU) and the Q-learning-based ensemble strategy. Section 4 presents a case study conducted at the Chongqing Tobacco Logistics Center, comparing the proposed method with single models and conventional ensemble techniques. Section 5 discusses limitations and future research directions, and Section 6 concludes the paper.

2. Literature Review

Traditional approaches to cable fire safety have primarily focused on fire detection and prevention. Although these techniques are well established, they are inherently reactive because alarms are typically issued only after observable environmental changes reach predefined criteria. For practical early warning, however, the key requirement is not only to identify hazardous conditions once they occur, but also to anticipate how risk develops over time. This shift in perspective motivates viewing cable fire risk prediction as a multivariate time series forecasting problem, in which future risk levels are inferred from historical sequences collected by electrical and environmental sensors. In parallel, data-driven artificial intelligence methods have demonstrated a strong capability in learning complex, nonlinear temporal relationships and have achieved robust performance in forecasting tasks across various application domains. Building on these methodological advances, this review summarizes research that is relevant to cable fire risk forecasting.

2.1. Cable Fire Detection Methods

Traditional cable fire detection methods primarily rely on temperature rise and the release of combustion gases. Standard detectors include point-type temperature detectors, point-type smoke detectors, cable-type temperature detectors, and suction-type smoke detectors [12]. In a full-scale cable trench test at a substation, Liu et al. [13] utilized fire dynamics simulation to analyze temperature fields, combustible gas concentrations, and alarm sequences across different detector types, highlighting the varying alarm characteristics of these detectors. Wang et al. [14] studied three-core power cables and demonstrated that sheath temperature can be used to infer inner-core overheating, thereby enabling earlier warnings. Han et al. [15] developed a SnO_2 nanofiber gas sensor that detects vapors from overheated PVC cables, supporting early alarms. In general, conventional methods detect fire by measuring environmental parameters and issuing alarms when predefined thresholds or rules are met [16]. However, such threshold-based schemes often produce false alarms, motivating the use of machine learning for more robust detection.

Machine learning-based fire detection approaches generally fall into two categories: (i) methods using fire-related physical parameters and (ii) image-based methods that identify flames or smoke. For the first category, Xiao et al. [17] used temperature, smoke concentration, and carbon monoxide as features for building fire detection. For the second category, image-based detection has advanced rapidly with the advent of deep learning, particularly convolutional neural networks (CNNs), Transformer-based models, and lightweight architectures. Li et al. [18] proposed a CNN-based framework for cable fire detection, which classifies images into three categories: fire, smoke, and non-fire. Sheng et al. [19] combined image segmentation and density-based spatial clustering (DBSCAN) with CNNs to improve fire and smoke identification accuracy.

Transformer architectures have been increasingly adopted because self-attention can model long-range dependencies in images [20,21]. Vuppari et al. [22] trained a Vision Transformer (ViT) for wildfire detection. Ahmad et al. [23] proposed CN2VF-Net, which combines CNNs for local feature extraction with a Transformer encoder to capture global relationships. Additionally, the practical deployment of models on resource-constrained platforms (e.g., drones and IoT devices) has driven research on lightweight models. SqueezeNet [24], EfficientNet [25], and MobileNet [26,27] are widely used as efficient backbones. Chang et al. [28] integrated MobileNetV3 into YOLOv5s to improve efficiency across multiple wildfire datasets. Jin et al. [29] proposed a lightweight model (SWVR) that combines a reparameterized Vision Transformer (RepViT) with a parameter-free attention module (SimAM) to enhance feature extraction at low computational cost.

2.2. Time Series Forecasting Methods

Time-series forecasting uses artificial intelligence to learn temporal dependencies and nonlinear relationships between a target variable and its influencing factors, enabling predictive decision-making in complex systems. Trend forecasting is essential for production planning, process control, and optimization, and can be effectively supported by time-series forecasting methods [30]. Zhou et al. [31] proposed a deep neural network that integrates traditional PID control with a BP neural network to improve water quality prediction for production decisions and risk reduction in aquaculture. Yan et al. [32] developed an LSTM-based method to predict the response time of umbilical cable top tension, enabling timely maintenance decisions in the event of extreme sea conditions. Shen et al. [33] presented an improved GRU model for forecasting financial indices (Hang Seng Index, German Stock Index, and S&P Index) and reported favorable performance across datasets. Li et al. [34] investigated multivariate time-series forecasting using CNNs and RNNs to capture short- and long-term dependencies, achieving promising results on multiple datasets.

Time-series forecasting has also been widely applied to predicting electrical parameters. Arikan et al. [35] measured dielectric parameters of aged high-voltage cables under overvoltage conditions and used artificial neural networks, decision trees, and curve fitting for interpolation and extrapolation. Ledari and Mirzaie [36] developed an artificial neural network to predict the maximum induced voltage on cable sheaths during lightning events. Thanh et al. [37] employed a GRU-based model to forecast leakage current on distribution system insulators using weather variables. Peng and Yu [38] proposed an LSTM-based method for predicting circuit current to enhance node-level current forecasting accuracy. However, existing studies on electrical parameter prediction typically focus on single-parameter thresholds, and few works leverage multidimensional risk factors to predict comprehensive cable fire risk.

2.3. Application of Hybrid Models

Single models often have limited representation capacity and generalization performance; therefore, hybrid and combination structures have been widely explored for time-series prediction. Pourhosseini et al. [39] developed a hybrid support vector machine model to predict total dissolved solids in rivers, enabling accurate water quality monitoring. Wang et al. [40] combined a convolutional neural network for feature extraction with the Informer model for long-sequence forecasting, proposing a wind power prediction method that exploits both local and long-range temporal patterns. Bai et al. [41] initialized neural network weights using a deep belief network and then applied reconstruction to obtain final predictions for daily reservoir inflow forecasting. Liu et al. [42] decomposed the wind speed series and used LSTM and Elman neural networks to forecast low- and high-frequency components, respectively. Sattarzadeh et al. [43] proposed a hybrid model that integrates an autoregressive moving average model with CNN and LSTM networks, achieving improved traffic flow prediction accuracy.

With the development of reinforcement learning, model integration strategies that adaptively combine multiple predictors have received increasing attention. Fu [44] proposed a reinforcement learning-based ensemble framework to learn the weights of integrated models for time-series forecasting. Li et al. [45] employed GRU networks to predict multiple $PM_{2.5}$ subsequences and then used Q-learning to combine them into final predictions. Tan et al. [46] evaluated temporal convolutional networks and LSTM models using the SARSA algorithm and implemented a reinforcement learning-based integrated model for $PM_{2.5}$ prediction. Overall, these studies suggest that reinforcement learning-driven ensembles can achieve strong predictive performance, motivating their application to comprehensive cable fire risk prediction.

In parallel with model-driven and data-driven forecasting, several studies have developed multi-indicator frameworks for assessing urban-scale fire risk. For historic urban centers, Ferreira et al. [47] combined physical and social indicators into a multi-vulnerability index to map and prioritize fire risk across city blocks in Valparaíso. Although such index-based approaches support spatial planning and mitigation, they typically rely on static or slowly varying indicators. Ferreira et al. [48] further proposed a simplified urban fire risk assessment method that integrates index-based analysis with GIS mapping to prioritize interventions and evacuation planning at the block scale. These urban-scale studies highlight the value of risk-informed decision support; in contrast, the present work addresses a complementary problem by focusing on cable-level risk prediction within indoor electrical infrastructure using rapidly evolving multivariate time-series data.

3. Methodology

The QBEL model proposed in this study forecasts future fire risk by adaptively weighting the outputs of three deep temporal base learners, including RNN, GRU, and LSTM, rather than using fixed combination weights. The relative predictive capability of different temporal networks can change across operating regimes. An adaptive weighting strategy can therefore enhance robustness and minimize systematic error as the data distribution evolves.

The model inputs are multidimensional sensor measurements including current (A), voltage (V), power (W), ambient temperature (°C), oxygen content (%), and relative humidity (%). These variables are measured by sensors installed in the transformer cabinet of a tobacco logistics center. The model output is a cable fire risk value computed by a system-generated risk value tailored to the single-core thin wire wrapped in green PVC flame-retardant material used at the study site. Physically, the risk value provides a composite proxy of early electrical fire development by integrating overload-related thermal stress in the conductor and insulation, the surrounding thermal environment, and combustion propensity in the cabinet atmosphere. As formulated in Equation (1), the risk value is obtained by mapping measured variables into monotonic, piecewise risk components and then aggregating them:

$$R_{\text{fire}} = \underbrace{R_I + R_V + R_P}_{\text{thermal stress}} + \underbrace{R_T}_{\text{environment}} + \underbrace{R_O + R_H}_{\text{combustion}} \quad (1)$$

The aggregation follows a dominance–collaboration logic, where the variable judged most critical under the current condition contributes more strongly, while other variables provide supportive modulation. When any key variable exceeds its recommended threshold, the corresponding component is amplified, resulting in a rapid escalation of the overall risk value. This reflects a conservative early warning principle consistent with engineering risk management practices. The intact on-site formulation generates the risk labels used for model training and testing. For confidentiality, the disclosed formulation and threshold rationale are provided in the Supplementary Materials, with sensitive information desensitized.

3.1. Framework of the QBEL Model

Figure 1 illustrates the QBEL framework, which consists of data preprocessing, base learning, and ensemble fusion. First, electrical operating variables, environmental conditions, and risk values within the same time frame are collected and normalized. The dataset is then split into training and testing sets and converted into supervised samples using a sliding window so that each input sequence corresponds to its future risk output. Next, RNN, GRU, and LSTM are trained as base learners on the training sequences to generate three risk forecasts. Finally, ensemble fusion is conducted as an iterative weight optimization process on the training set, where the combination weights are updated to

reduce the fusion loss, and the resulting weights are used to fuse the three base forecasts for the final risk prediction.

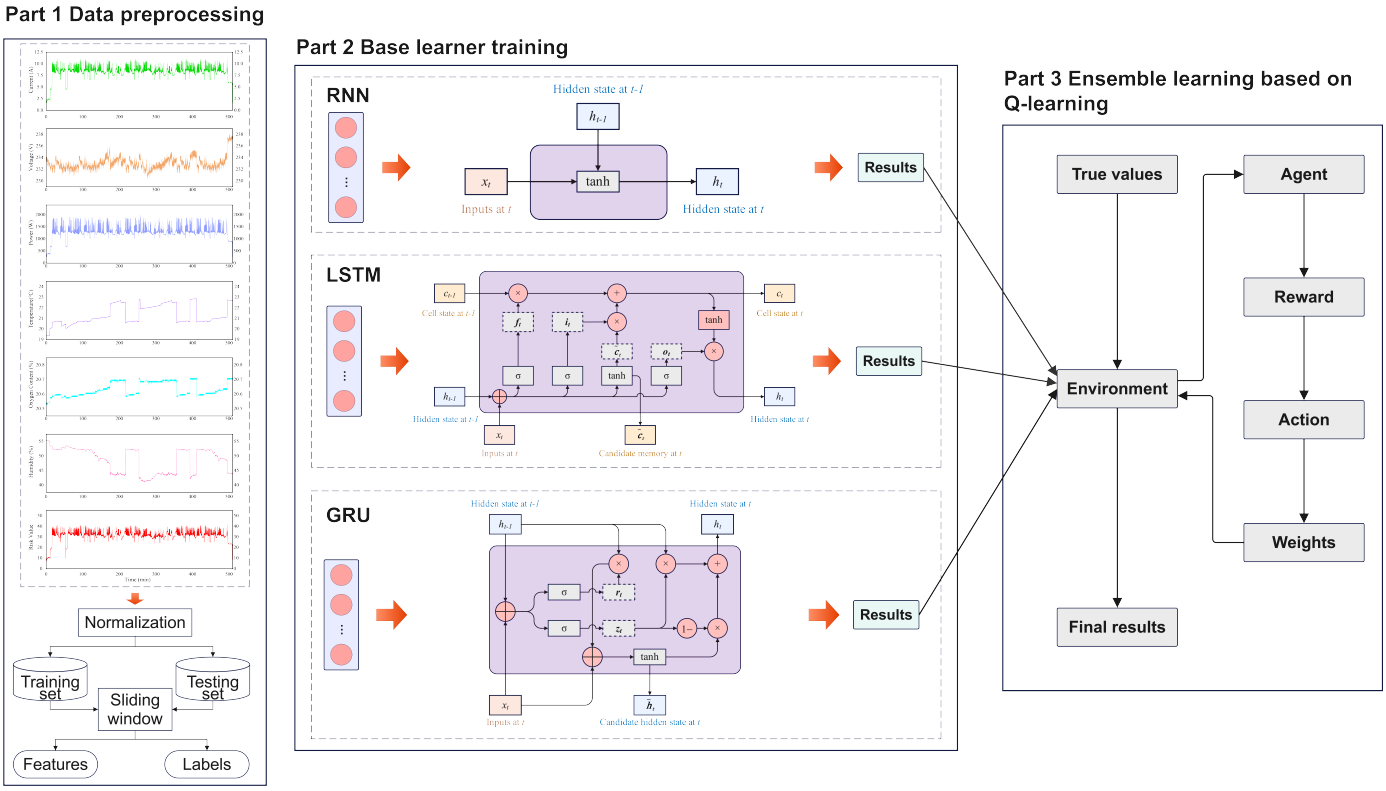

Figure 1. Framework of the QBEL model; For detailed explanations of model modules, refer to Section 3.2.

In particular, the fusion stage is formulated as a reinforcement learning task. The environment is defined by the base learner predictions together with the corresponding ground truth risk values, and the agent is the Q-learning module that updates the fusion weights. The state is the current weight vector assigned to the three base learners. The action is selected from a discrete set of weight adjustment patterns that increase or decrease the three weights by a fixed step and then normalize them. The reward is derived from the reduction in mean squared error after applying the action. Through repeated updates, the agent learns a weighting policy that yields a lower fusion loss than fixed-weight combinations.

3.2. Base Learners Based on Deep Learning

3.2.1. Recurrent Neural Network (RNN)

Early neural network models, like the multilayer perceptron, used for handling independent inputs and outputs, featured fully connected structures between layers. RNN is an extension of these early network models [49,50]. As illustrated in Figure 2, RNN introduces lateral connections between hidden layer units, resembling a chain of recurrent units compared to a multilayer perceptron. Through feedback connections, the hidden layer neurons process independent inputs and outputs, leveraging the input from the current time step and the hidden state from the previous time step to generate the output for the current time step and a new hidden state. This enables capturing temporal dependencies in sequential data, simultaneously grasping dependencies between elements and contextual information.

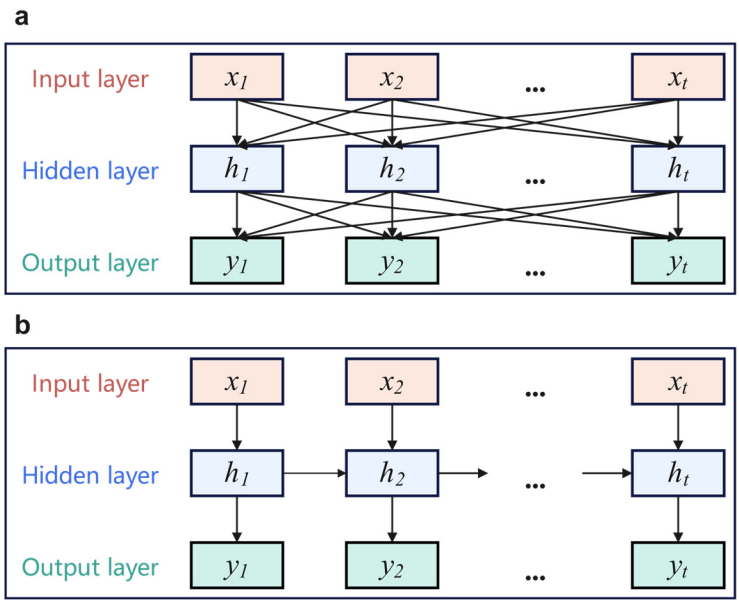

Figure 2. Structure of (**a**) multi-layer perceptron; (**b**) recurrent neural network. t represents the t-th timestamp in the time series, x_t represents the input at time t, h_t represents the hidden state at time t, and y_t represents the fire risk value of the system at time t; The "..." denote omitted time steps, included to simplify the visualization of the network structure.

The input layer represents the sample features of the network input. In this study, the input of each time step includes current, voltage, power, ambient temperature, humidity, oxygen content, and risk values from the alert system.

The internal structure of an RNN unit is typically characterized by its recurrent connections. Each unit takes inputs and hidden states from the previous time step, processes them through an activation function, and outputs a new hidden state. This hidden state is then fed into the next unit in the sequence and contributes to generating predictions or outputs. By utilizing this feedback loop to store and utilize past information, RNN effectively addresses sequence problems. When using RNN as a base learner for cable fire risk prediction, historical data serves as samples, and the multidimensional risk features at each time step act as inputs. Through Equation (2), the hidden state is updated, storing all risk sequence information in the final hidden state.

$$h_t = \tanh(w[h_{t-1}, x_t] + b) \tag{2}$$

where tanh represents the hyperbolic tangent function, w is the weight parameter of the model, and b is the bias parameter.

3.2.2. Long Short-Term Memory Network (LSTM)

LSTM represents a significant advancement in the field of RNN, addressing the problem of vanishing or exploding gradients during the training process of RNN networks, thereby preventing hindrance to effective learning in recurrent networks. As a variant of RNN network architecture, LSTM incorporates a distinctive feature known as threshold structures [51]. These threshold structures comprise three fundamental components: the input gate, the output gate, and the forget gate, as illustrated in Figure 3. The input gate regulates the impact of new input information on the current time step, determining how much new information will be added to the cell state. The forget gate determines how much information from the cell state of the previous time step will be discarded, specifying which information in the cell state should be forgotten. The output gate controls the influence

of information in the cell state on the hidden state at the current time step, determining which information in the cell state will be output to the hidden state at the current time step. These gates manage the flow and retention of information, selectively remembering or forgetting crucial details in the sequence.

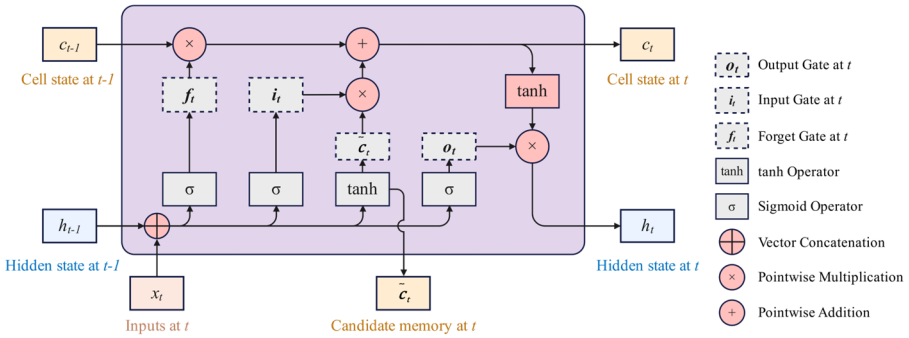

Figure 3. Structure of a hidden layer unit in LSTM.

In this study, LSTM is employed as one of the base learners. During the training process, the network weights are automatically updated when multidimensional risk features and corresponding fire risk values are input. This dynamic adjustment of the overall proportion at each time step is achieved while maintaining consistent model parameters. The input gate, output gate, and forget gate collectively determine the extent to which risk features from the historical sequence are transmitted and forgotten, ultimately influencing the final fire risk prediction. Equations (3)–(8). outlined below summarizes the computation process for updating the hidden state.

$$i_t = \sigma\left(w_i[h_{t-1}, x_t] + b_i\right) \tag{3}$$

$$f_t = \sigma\left(w_f[h_{t-1}, x_t] + b_f\right) \tag{4}$$

$$\tilde{C}_t = \tanh(w_c[h_{t-1}, x_t] + b_c) \tag{5}$$

$$C_t = f_t \bigodot C_{t-1} + i_t \bigodot \tilde{C}_t \tag{6}$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \tag{7}$$

$$h_t = o_t \bigodot \tanh(C_t) \tag{8}$$

where i_t, f_t and o_t represent input gate, forget gate and output gate respectively, $\tilde{C}_t$ and C_t represent candidate memory cell and memory cell respectively, σ represents the sigmoid function, $\bigodot$ represents element level multiplication or Hadamard product, w_f w_i w_c w_o are the weight parameters of the model, and b_f b_i b_c b_o are the bias parameters.

3.2.3. Gated Recurrent Unit Network (GRU)

GRU represents another variant of the RNN network structure. Compared with other recurrent neural networks, GRU shares many similar characteristics but also introduces some significant differences. GRU is designed to address the challenges of vanishing and exploding gradients while providing a simplified gating mechanism for more effective sequence modeling [52].

As shown in Figure 4, similar to LSTM, GRU incorporates a gating structure, but unlike the three gates of LSTM, GRU utilizes only two gates: the update gate and the reset gate. The update gate determines how the hidden state at the current time step combines with the hidden state from the previous time step and the current input, while the reset gate determines how the hidden state from the previous time step influences the input at the

current time step. These gating mechanisms enable the GRU network to selectively update and reset information when processing sequence data, allowing it to adapt autonomously and choose how to update and reset information based on the same model parameters. This adaptive mechanism helps GRU better capture long-term dependencies in sequences.

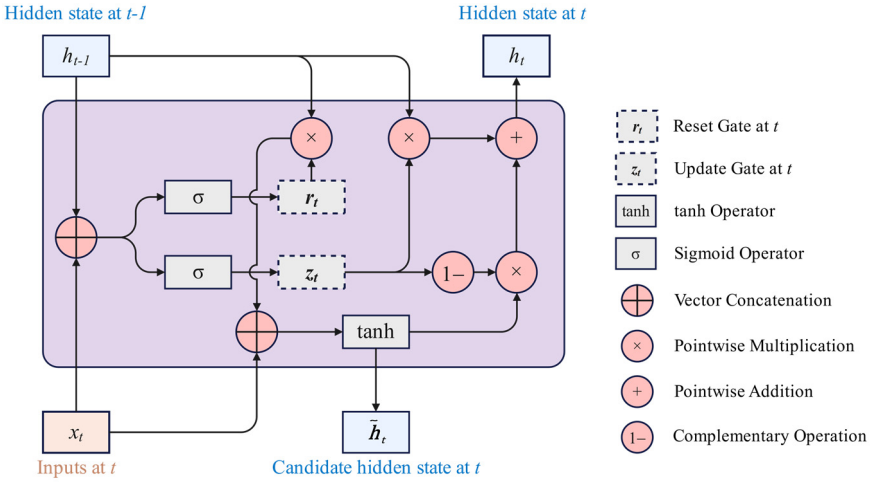

Figure 4. Structure of a hidden-layer unit in GRU.

The training process involves several steps. First, the hidden state and other model parameters are initialized. Second, for each time step, the update gate and reset gate results are computed based on the multidimensional risk features of the input and the hidden state from the previous time step. Third, the temporary hidden state is calculated using the update gate and reset gate. Fourth, the final hidden state is computed based on the temporary hidden state and the update gate. Finally, the hidden state is used to generate the ultimate prediction for the fire risk state. The computation processes for updating the hidden state are outlined in Equations (9)–(12).

$$z_t = \sigma(w_z[h_{t-1}, x_t] + b_z) \tag{9}$$

$$r_t = \sigma(w_r[h_{t-1}, x_t] + b_r) \tag{10}$$

$$\tilde{h}_t = tanh\left(w_{\tilde{h}}\left[r_t \bigodot h_{t-1}, x_t\right]\right) \tag{11}$$

$$h_t = (1 - z_t) \bigodot h_{t-1} + z_t \bigodot \tilde{h}_t \tag{12}$$

where z_t and r_t represent update gate and reset gate respectively, $\tilde{h}_t$ represents intermediate hidden state, w_r, w_z, $w_{\tilde{h}}$ are the weight parameters of the model, b_r, b_z are the bias parameters.

3.3. Ensemble Learning Based on Q-Learning

The ensemble aims to learn an online weight-update policy so that each base predictor's contribution can adapt over time to its recent predictive performance. This is implemented using tabular Q-learning, where the action–value function $Q(s, a)$ is updated via temporal-difference learning with ε-greedy exploration. Unlike conventional adaptive weighting, which applies a fixed error-to-weight mapping at each step, the proposed approach formulates weight adjustment as a sequential decision process. It optimizes cumulative predictive improvement under non-negativity and sum-to-one weight constraints.

Figure 5 summarizes the overall QBEL workflow, including (i) training the base predictors on sliding-window samples, and (ii) constructing the Q-learning loop that maps recent-window error "extent labels" to weight-update actions, iteratively updating the Q-table and producing time-varying ensemble weights.

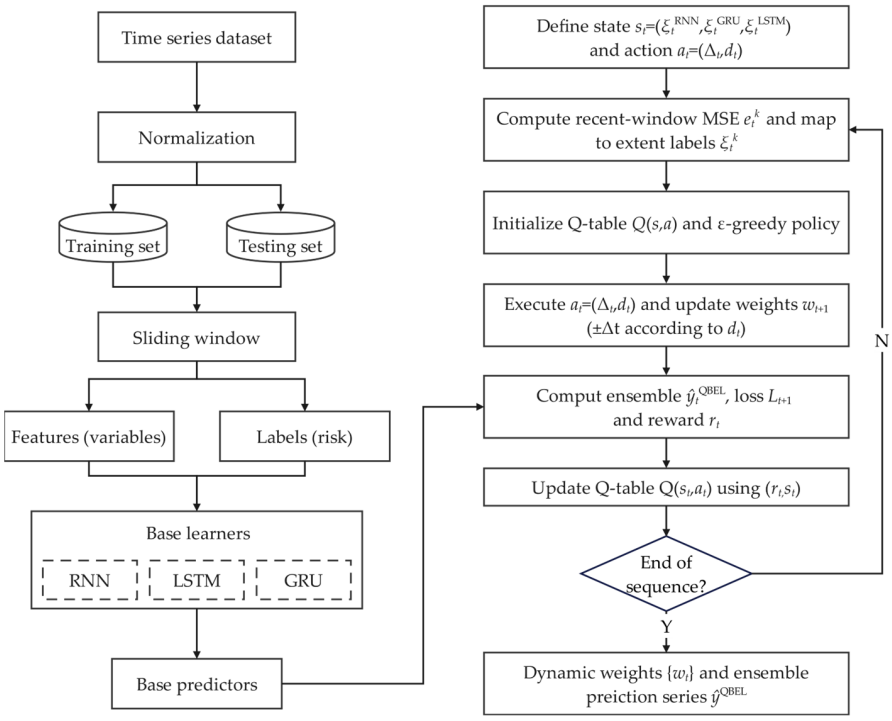

Figure 5. Roadmap of the QBEL model.

3.3.1. Ensemble Learning Model

At each time t, the three base learners ($k \in \{\text{RNN, GRU, LSTM}\}$) provide one-step-ahead forecasts $\hat{y}_t^{\text{RNN}}, \hat{y}_t^{\text{GRU}}, \hat{y}_t^{\text{LSTM}}$. The ensemble forecast is computed as a convex combination:

$$\hat{y}_t^{\text{QBEL}} = \sum_k w_t^k \hat{y}_t^k, \;\; w_t^k \geq 0, \; \sum_k w_t^k = 1 \tag{13}$$

where $w_t = \left(w_t^{\text{RNN}}, w_t^{\text{GRU}}, w_t^{\text{LSTM}}\right)^\top$ denotes the time-varying weight vector. The adaptation problem is therefore not to identify a single global weight vector, but rather to produce a sequence $\{w_t\}$ that tracks temporal changes in the predictors' relative reliability, thereby improving short-horizon predictive accuracy.

3.3.2. State and Action Design

The adaptive fusion process is formulated as a Markov decision process (MDP) in which an agent adjusts the ensemble weights sequentially over time. At each time step t, the agent observes a state s_t, selects a weight-adjustment action a_t, updates the weight vector, and receives a reward after the next-step ground truth becomes available.

State. The state s consists of a series of discrete extent labels ξ that describe the recent-window prediction errors of each base learner. To capture temporal dependence in the time series, a sliding window of length l is introduced to define the local time frame over which recent performance is evaluated. Accordingly, at each time step t, the recent-window mean square error (MSE) of base learner k is calculated as:

$$\bar{e}_t^k = \frac{1}{l} \sum_{j=t-l+1}^{t} \left(y_j - \hat{y}_j^k\right)^2 \tag{14}$$

As t increases, this produces an error sequence for each predictor $\left\{\bar{e}_t^k\right\}$. To obtain a compact, tabular state space, each $\bar{e}_t^k$ is mapped to an extent label ξ_t^k using percentile partitioning on the training set. Specifically, for each predictor, we compute nine percentile thresholds from the training-set values of $\left\{\bar{e}_t^k\right\}$:

$$\xi_t^k = \begin{cases} 1, & \bar{e}_t^k \le p_1^k \\ 2, & p_1^k < \bar{e}_t^k \le p_5^k \\ \vdots & \\ 10, & \bar{e}_t^k > p_{99}^k \end{cases}, \quad P^k = \left\{ p_1^k, p_5^k, p_{10}^k, p_{25}^k, p_{50}^k, p_{75}^k, p_{90}^k, p_{95}^k, p_{99}^k \right\} \tag{15}$$

where p_q^k denotes the q-th percentile of the distribution of $\bar{e}_t^k$, which partitions the error range into 10 ordered levels.

Accordingly, the overall state s_t used by Q-learning at the time step t is the triplet of extent label:

$$s_t = \left(\xi_t^{RNN}, \xi_t^{GRU}, \xi_t^{LSTM} \right) \tag{16}$$

Action. An action is defined as a pair $a_t = (\Delta_t, d_t)$, consisting of a step size Δ_t and a direction mask d_t:

$$\Delta_t \in \{0.001, 0.005, 0.01, 0.05, 0.1\} \tag{17}$$

$$d_t = \left(d_t^{\mathrm{RNN}}, d_t^{GRU}, d_t^{LSTM} \right) \in \{0,1\}^3 \tag{18}$$

The direction mask indicates which weights should increase ($d_t^k = 1$) and which should decrease ($d_t^k = 0$). Define two index sets: $I_t^+ = \left\{ k : d_t^k = 1 \right\}$, $I^- = \left\{ k : d_t^k = 0 \right\}$. If both sets are nonempty, the update transfers a total amount Δ_t from the "decrease" group to the "increase" group:

$$w_t^k = \begin{cases} w_t^k + \frac{\Delta}{|I^+|}, & k \in I^+ \\ w_t^k - \frac{\Delta}{|I^-|}, & k \in I^- \end{cases} \tag{19}$$

If $d_t = (0,0,0)$ or $d_t = (1,1,1)$, no meaningful increase or decrease split exists; therefore, these two patterns are treated as "no-change" actions and the weights remain unchanged. After the tentative update, weights are adjusted to remain valid: negative values are set to zero.

3.3.3. Reward, Q-Table Update, and Deployment

Loss and reward. After executing action a_t, the updated weights w_{t+1} are used to form the ensemble prediction $\hat{y}_{t+1}^{\mathrm{QBEL}}$, and the agent would receive a reward proportional to the reduction in loss.

The one-step prediction loss $\mathcal{L}_t$ and reward r_t are defined as:

$$\mathcal{L}_t = \left(y_t - \hat{y}_t^{\mathrm{QBEL}} \right)^2 \tag{20}$$

$$r_t = M(\mathcal{L}_t - \mathcal{L}_{t+1}) \tag{21}$$

where M is a coefficient that amplifies minor loss differences so that the agent remains sensitive to incremental improvements that are important for early risk detection. Under this definition, $r_t > 0$ indicates that the action reduces the prediction error, whereas $r_t < 0$ indicates deterioration.

Q-table update. During training, actions are selected using an ε-greedy strategy. After observing reward r_t and the next state s_{t+1}, the Q-table is updated by:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \tag{22}$$

where α is the learning rate and γ is the discount factor.

Deployment. During testing, the same percentile thresholds learned from the training set are used to map recent-window MSE values to extent labels ξ_t. At each time step, the agent selects the action a_t with the largest $Q(s_t, a)$, updates weights accordingly, and

outputs the ensemble prediction $\hat{y}_t^{\mathrm{QBEL}}$. As a result, the ensemble weights vary over time in response to changes in the recent predictive performance of the base learners.

4. Case Study

4.1. Cable Fire Risk Time Series Dataset

This study utilized recorded data from the intelligent fire alarm system installed at the Chongqing Tobacco Logistics Distribution Center in China to validate the proposed model. The system provides real-time measurements of electrical operating parameters and environmental parameters for the distribution line in the distribution room, together with a fire risk value reported by the alarm platform. The dataset comprises 3060 time-series records collected on 15 March 2022, representing a typical working day. The sampling interval is 10 s, covering approximately 8.5 h of operation from 9:00 to 17:30. Each record includes seven variables: current, voltage, power, ambient temperature, relative humidity, oxygen content, and the risk value.

Figure 6 presents the time histories of the monitored variables and the corresponding risk value. The electrical variables exhibit a clear start-up phase, followed by a prolonged period of quasi-steady operation with intermittent fluctuations. Current increases rapidly from about 0 to approximately 10 A, then remains mainly within about 8 to 11 A with occasional short drops to around 5 A. Voltage is relatively stable in the range of roughly 231 to 235 V, with a brief increase approaching 237 V near the end of the record. Power rises quickly to a typical operating level of approximately 1000 to 1700 W, with frequent short spikes reaching around 2000 W and occasional dips to roughly 600 to 800 W. The environmental variables vary more slowly but exhibit step-like changes, with ambient temperatures ranging from approximately 19 to 23 °C, relative humidity between about 41% and 55%, and oxygen content between about 20.55% and 20.70%. The risk value increases sharply in the early period and then remains predominantly within a range of about 28 to 36, with frequent short-duration spikes approaching 40 and occasional abrupt reductions. These patterns indicate nonstationary behavior, characterized by both stable segments and rapid excursions, which motivates the use of temporal base learners and an adaptive fusion strategy that remains stable even when operating conditions shift.

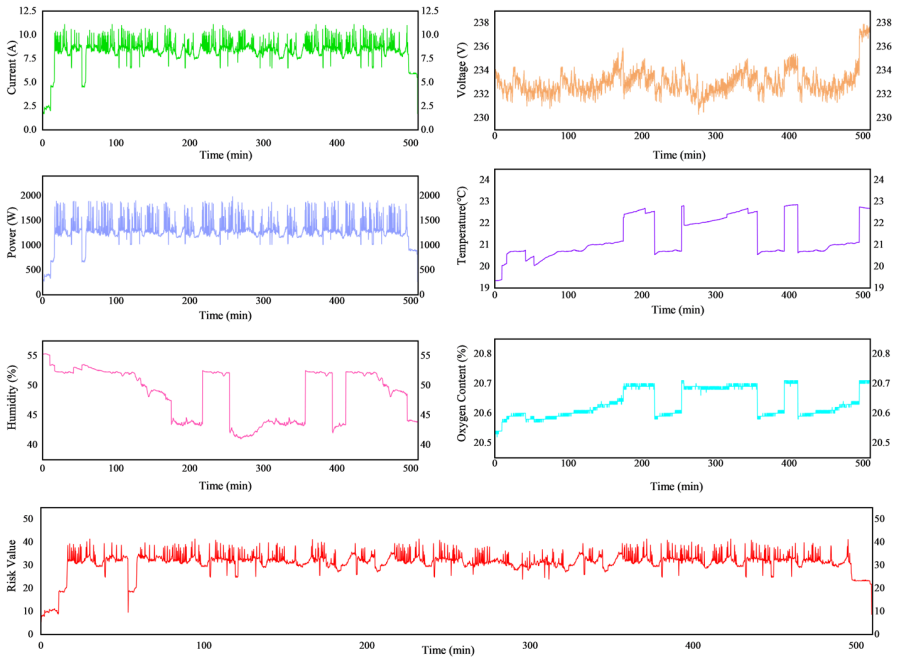

Figure 6. Original cable fire risk time series data.

The cable fire risk prediction model in this paper was trained using the PyTorch deep learning framework, an open-source tool. All computations were performed on a Dell Precision 7920 server running Windows Server 2019. The server was equipped with an Intel® Xeon® Gold 6230 CPU clocked at 2.1 GHz and 64 GB of memory, as well as an NVIDIA GeForce RTX 3080 GPU with 10 GB of memory, 8704 CUDA cores, and 29.77 TFLOPs of FP32 performance. The software environment for the experiment included Anaconda 4.10.1, a scientific computing package; PyTorch 1.8.1, a deep learning framework; and Python 3.8. The training process was accelerated using CUDA 11.1 for GPU acceleration.

4.1.1. Data Preprocessing

In this study, each recorded instance in the data was treated as a model feature, denoted as x_t, as shown in Equation(23), where t represents the specific moment. To avoid the dominance of certain variables, the dataset underwent normalization, scaling the values of each feature to a standard range. Taking the current as an example, the normalization formula is shown in Equation (24).

$$x_t = (I_t, U_t, P_t, O_t, T_t, H_t, R_t) \tag{23}$$

$$I_i' = \frac{I_i - \min(I)}{\max(I) - \min(I)} \tag{24}$$

where $I_t, U_t, P_t, O_t, T_t, H_t, R_t$ represent the input features at time t, which correspond to current, voltage, power, oxygen content, temperature, humidity, and system risk value, respectively.

The training set and testing set were divided in an 8:2 ratio, and the data were processed using a sliding window approach, which separates the time series data into segments and extracts features. To prepare the training data for time series analysis, input-output pairs were generated within the data, considering a specified time step, denoted as l. For each sample in the dataset, an input feature matrix X_t and the corresponding target variable y_t were constructed, as shown in Equations (25) and (26).

$$X_t = \begin{bmatrix} x_{t-l+1} & x_{t-l+2} & \cdots & x_t \end{bmatrix}^{\mathrm{T}} \tag{25}$$

$$y_t = R_{t+1} \tag{26}$$

where t represents the t-th timestamp in the time series, X_t represents the input at time t, and y_t represents the label used for training the model corresponding to X_t.

All models evaluated in Sections 4.2 and 4.3 share the same data preprocessing pipeline and data split. The raw data were normalized feature-wise and converted into supervised samples using sliding windows with a length of $l = 10$ s. An 8:2 chronological split is adopted: the first 80% of the sequence is used for training and the remaining 20% for testing, without any shuffling across timestamps to avoid information leakage. All baseline models, the three base learners (RNN/LSTM/GRU), and the proposed QBEL ensemble are trained on the same training set and evaluated on this fixed test set.

4.1.2. Performance Evaluation Metrics

In this study, we employed four commonly used metrics to assess the model's performance, including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE).

MSE and RMSE eliminate the influence of error signs, providing a comprehensive assessment of the overall fluctuation in prediction errors. MAE evaluates the average difference between the model's predicted results and the actual values, representing the

average magnitude of errors in typical cases. MAPE measures the relative error between the model's predicted results and the ground truth in percentage, thereby assessing the model's relative accuracy. These metrics are shown in Equations (27)–(30)

$$\mathrm{MSE} = \frac{1}{N}\sum_{m=1}^{N}(y(m) - \hat{y}(m))^2 \tag{27}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{m=1}^{N}(y(m) - \hat{y}(m))^2} \tag{28}$$

$$\mathrm{MAE} = \frac{1}{N}\sum_{m=1}^{N}|y(m) - \hat{y}(m)| \tag{29}$$

$$\mathrm{MAPE} = \frac{1}{N}\sum_{m=1}^{N}\left|\frac{y(m) - \hat{y}(m)}{y(m)}\right| \times 100\% \tag{30}$$

where $y(m)$ represents the actual risk value and $\hat{y}(m)$ denotes the predicted risk value, and N is the number of samples in $y(m)$.

Considering these four metrics collectively enables a comprehensive assessment of the model's predictive ability, providing insights into its performance from different perspectives. Utilizing these metrics facilitates the provision of quantitative evaluation results to determine the model's applicability and effectiveness.

4.1.3. Training Hyperparameters

To facilitate reproducibility of the proposed QBEL model and its base learners, we report the principal training hyperparameters (see Table 1), including learning rate (α), discount factor (Υ), and probability of exploration (ε). Residual connections were added to the hidden layers of the base learners to alleviate gradient vanishing during training, and a dropout rate of 0.2 was used to reduce overfitting to noise in the time-series data.

To compare the proposed Q-learning-based ensemble with conventional ensemble strategies, two types of baselines were constructed: boosting and stacking, both built on the same three base learners (RNN, LSTM, and GRU). For promoting, XGBoost with classification and regression trees (CART) as weak learners was adopted. At each time step, the input to XGBoost consisted of the predictions from the three base learners, and the target was the corresponding ground truth. Models were trained with numbers of trees $n \in \{100, 200, 300\}$ and maximum depths $d \in \{3, 6, 10\}$. For stacking, a meta-learner was trained on the combined predictions of the three base learners. The meta-learner was instantiated as linear regression (LR), random forests ($n \in \{100, 200, 300\}$, $d \in \{4, 8, 12\}$), or XGBoost ($n \in \{100, 200, 300\}$, $d = 3$). For all boosting and stacking baselines, meta-learners were trained on the base-learner predictions from the training portion of the data and evaluated on the same train-test split as QBEL, ensuring a direct and fair comparison.

To further assess the effectiveness of QBEL, several recent time-series forecasting models were considered: Dlinear [53], PatchTST [54], MoLE [55], and Fredformer [56]. For all these models, the input is the same normalized multivariate time series with 7 features, represented by a sliding window of length $l = 10$ s, and the same 8:2 chronological train-test split is used. DLinear follows a linear decomposition-based design. PatchTST is a Transformer variant that uses patch embeddings to capture long-range dependencies. MoLE adopts a mixture-of-linear-experts architecture to increase representational capacity. Fredformer incorporates frequency-domain decomposition into a Transformer backbone for spectral correction. To evaluate model sensitivity, several hyperparameter settings were

considered: PatchTST with patch lengths $p \in \{8, 16, 32\}$; MoLE with numbers of experts $e \in \{4, 8, 16\}$; and Fredformer with bandwidth parameter pairs $(b, w) \in \{(3, 2), (2, 3), (6, 1)\}$.

Table 1. Model hyperparameter settings of QBEL model.

Model	Hyperparameter	Value
RNN/GRU/LSTM		
Network structure	Hidden Layers	2
	Hidden Units	256/128/128
	Dropout Rate	0.2
Training Settings	Time Step Length	10
	Learning Rate (α)	0.001
	Batch Size	64
	Epoch	200
Optimization	Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
QBEL		
	Reward Multiplier (M)	100
	Learning Rate (α)	0.001, 0.01, 0.1
	Discount Factor (Υ)	0.7, 0.8, 0.9
	Probability of Exploration (ε)	0.05, 0.1, 0.5
	Maximum steps for a single episode	2000
	Maximum episodes	10

4.2. Model Comparison

4.2.1. QBEL vs. Conventional Ensemble Methods

QBEL was benchmarked against representative boosting and stacking ensembles (see Table 2). The three base learners (RNN, LSTM, GRU) were observed to behave consistently across runs (MSE std. < 0.24; time std. < 0.01 s; memory std. < 1 MB), and their mean results were adopted as the baseline.

XGBoost was found to yield weaker and more volatile accuracy (mean MSE > 20.82; std. > 6.75), with RAM usage typically exceeding 200 MB on the testing set, and a pronounced train-test gap was observed. These outcomes can be partly explained by the fact that, even when time-lagged windows are used as input features, tree-based ensembles still treat samples as conditionally independent. As a result, they struggle to capture smooth temporal evolution and long-range dependencies beyond the fixed window, and they tend to overfit to local patterns in this relatively small dataset [57]. In contrast, stacking was observed to improve accuracy (mean MSE $\approx$ 1.85; 27.14% below baseline), with deeper meta-learners generally associated with further gains; however, these gains were accompanied by substantially increased computational demands (inference time +308.72%; GPU memory +97.66%) and notable sensitivity to meta-learner choice and hyperparameters (MSE 1.33 to 2.95; std. 0.58). This sensitivity implies additional tuning effort for industrial deployment and heavier resource demands on edge devices.

QBEL achieved both high accuracy and stability across hyperparameter settings (mean MSE $\approx$ 1.83; MSE std. < 0.24) with modest resource growth (time +23.59% on average; GPU memory +40.46%), markedly lower than stacking. Robustness can be attributed to the MDP formulation: states were defined to encode real-time errors and temporal risk trends; rewards were designed to balance immediate error reduction with smooth weight transitions; and actions were restricted to interpretable, bounded updates [58]. Through this structure, sensitivity to transient noise is curtailed, and the volatility commonly introduced by meta-models can be considerably alleviated. For subsequent analyses, the best-performing QBEL configuration ($\alpha = 0.001$, $\Upsilon = 0.9$, $\varepsilon = 0.10$) was adopted.

Table 2. Performance comparison of ensemble methods by accuracy and computational metrics.

Model	Accurate Metrics				Computation Metrics	
	MSE	**RMSE**	**MAE**	**MAPE (%)**	**Time (s)**	**Memory (MB)**
Base learner						
RNN	2.8158 *	1.6780 *	1.2687 *	4.1136 *	0.0641	31.8054
LSTM	2.5668 *	1.6021 *	1.1095 *	3.7033 *	0.0744	33.5121
GRU	**2.2242 ***	**1.4914 ***	**0.9456 ***	**3.0627 ***	**0.0691**	**31.2892**
Average base learners	2.5356 *	1.5905 *	1.1079 *	3.6265 *	0.0692	32.2022
Boosting						
XGboost (n = 100, d = 3)	21.5154	4.3791	1.6763	3.0605	**0.1219**	**118.1680**
XGboost (n = 100, d = 6)	33.3831	5.7778	2.1889	4.0715	0.1610	120.4141
XGboost (n = 100, d = 10)	34.3907	5.8644	2.1833	4.0487	0.3027	128.0234
XGboost (n = 200, d = 3)	19.0045	4.3594	1.7309	3.2193	0.1378	118.8281
XGboost (n = 200, d = 6)	32.7201	5.7201	2.1315	3.9570	0.2487	123.4922
XGboost (n = 200, d = 10)	34.1319	5.8314	2.1467	3.9736	0.4852	133.3594
XGboost (n = 300, d = 3)	**17.6615**	**4.2026**	**1.6451**	**3.0546**	0.1573	120.0430
XGboost (n = 300, d = 6)	32.4869	5.6997	2.1111	3.9164	0.2607	126.0977
XGboost (n = 300, d = 10)	34.0937	5.8353	2.1445	3.9697	0.8686	141.3555
Stacking						
Stacking-LR	1.9327 **	1.3902 **	1.1125 **	3.8710 **	**0.0821**	37.0147
Stacking-RF (n = 100, d = 4)	2.9452	1.7161 *	1.1274 *	4.2935	0.2193	36.2608
Stacking-RF (n = 100, d = 8)	1.5108	1.2291 *	0.7665 *	2.9222 *	0.2221	**35.2561**
Stacking-RF (n = 100, d = 12)	1.3459	1.1601 *	0.6480	2.5623	0.2261	40.8975
Stacking-RF (n = 200, d = 4)	2.8290	1.6820 *	1.1110 *	4.2080 *	0.3212	36.5108
Stacking-RF (n = 200, d = 8)	1.4975	1.2237 *	0.7626 *	2.8968 *	0.3393	36.8678
Stacking-RF (n = 200, d = 12)	**1.3327**	**1.1544 ***	**0.6453**	**2.5408**	0.3361	45.9874
Stacking-RF (n = 300, d = 4)	2.8280	1.6817 *	1.1076 *	4.2052 *	0.4269	36.7334
Stacking-RF (n = 300, d = 8)	1.4986	1.2242 *	0.7633 *	2.9035 *	0.4392	41.4991
Stacking-RF (n = 300, d = 12)	1.3335	1.1548 *	0.6454	2.5456	0.4487	51.4952
Stacking-XGboost (n = 100, d = 3)	1.7434	1.3218 **	0.8487 **	3.1442 *	0.1850	141.7608
Stacking-XGboost (n = 200, d = 3)	1.6407	1.2809 *	0.8049 *	3.0066 *	0.2059	143.0069
Stacking-XGboost (n = 300, d = 3)	1.5795	1.2568 *	0.7742 *	2.9137 *	0.2249	144.1827
Q-learning (this work)						
QBEL (α = 0.001, Υ = 0.7, ε = 0.05)	2.6096 **	1.6154 ***	1.2783 ***	3.9904 **	0.0890	46.3506
QBEL (α = 0.001, Υ = 0.8, ε = 0.10)	1.7457 **	1.3212 **	0.9040 **	2.7749 **	0.0845	**44.9795**
QBEL (α = 0.001, Υ = 0.9, ε = 0.10)	**1.7287 ** **	**1.3148 ** **	**0.8352 ** **	**2.6620 ** **	0.0845	45.1514
QBEL (α = 0.001, Υ = 0.9, ε = 0.50)	1.7405 **	1.3193 **	0.8825 **	2.7065 **	0.0844	45.1397
QBEL (α = 0.001, Υ = 0.7, ε = 0.30)	1.7358 **	1.3175 **	0.8762 **	2.6737 **	0.0844	45.2334
QBEL (α = 0.010, Υ = 0.8, ε = 0.05)	1.9542 **	1.3979 **	1.0149 ***	3.1437 **	0.0852	45.1280
QBEL (α = 0.010, Υ = 0.9, ε = 0.10)	1.7290 **	1.3149 **	0.8882 **	2.7230 **	0.0867	45.0030
QBEL (α = 0.010, Υ = 0.7, ε = 0.50)	1.7456 **	1.3212 **	0.8782 **	2.6929 **	0.0847	45.2999
QBEL (α = 0.010, Υ = 0.8, ε = 0.30)	1.7384 **	1.3185 **	0.8982 **	2.7594 **	**0.0838**	45.2100
QBEL (α = 0.100, Υ = 0.9, ε = 0.05)	1.7481 **	1.3221 **	0.9010 **	2.7674 **	0.0842	45.1084
QBEL (α = 0.100, Υ = 0.7, ε = 0.10)	1.7344 **	1.3170 **	0.8925 **	2.7347 **	0.0869	45.1475
QBEL (α = 0.100, Υ = 0.8, ε = 0.50)	1.7292 **	1.3150 **	0.8869 **	2.7184 **	0.0890	45.0147

Note: the symbol of (*) denotes the relative train-test discrepancy of the primary accuracy metric: * indicates <100%, ** indicates <50%, and *** indicates <10%. "Time (s)" indicates inference time on the test set (n = 612). "Memory (MB)" refers to GPU memory for base learners and QBEL models, as well as RAM usage for boosting and stacking models. The bolded values in this table represent the best performance for the corresponding metrics.

4.2.2. QBEL vs. Base Learners

To evaluate the effectiveness of each prediction method, four different approaches, RNN, LSTM, GRU, and QBEL, were employed to develop multivariate prediction models for cable fire risk. The prediction results of these four models on a subset of samples from the testing set, along with the actual system risk values, are shown in Figure 7.

It can be concluded that all models can predict the fire risk of the system, and their performance varies at different time steps. However, the QBEL model fits the actual data most closely overall. This may be attributed to the fact that the threshold structure within the hidden units of LSTM and GRU networks discards and retains different information from the input sequences. In contrast, the QBEL model makes decisions by combining these three methods, thereby enhancing its adaptability to other sequences and achieving more accurate predictions.

According to the results of Table 2 mentioned above, it is evident that the models consistently demonstrate a pattern regarding MSE, RMSE, MAE, and MAPE, with errors ranked from largest to smallest: RNN, GRU, LSTM, QBEL. Compared to individual models, the ensemble model demonstrated a significant reduction in error across all four metrics. The improvement Percentages of MSE (P_{MSE}), improvement Percentages of RMSE (P_{RMSE}), improvement Percentages of MAE (P_{MAE}), and improvement Percentages of MAPE (P_{MAPE}) were utilized to evaluate the performance improvement of the QBEL model in reducing errors. The extent of error reduction for the proposed model compared to individual models is demonstrated in Table 3, which showcases a significant enhancement in all four metrics, underscoring its superiority.

The effectiveness of different forecasting methods is depicted in Figure 8. Across the validation horizon, other models are observed to produce errors with distinct temporal patterns, reflecting the non-stationary evolution of the system rather than a deterministic dependence on absolute time. The RNN tends to overpredict, whereas the GRU and LSTM tend to underpredict, and their errors complement one another at different moments. In contrast, QBEL is observed to confine errors within a narrower band and to suppress significant excursions, yielding more minor deviations for most time points.

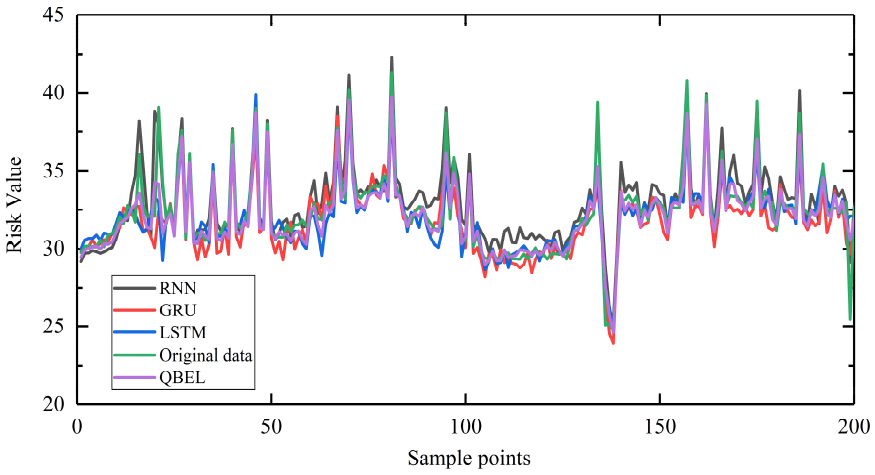

Figure 7. Prediction results of four models on a subset of the testing set.

Table 3. Improvement of the QBEL model relative to individual models.

Compared Models	P_{MSE} (%)	P_{RMSE} (%)	P_{MAE} (%)	P_{MAPE} (%)
RNN	38.6	21.6	34.3	32.8
GRU	32.7	17.9	24.7	25.4
LSTM	22.3	11.8	11.7	9.8

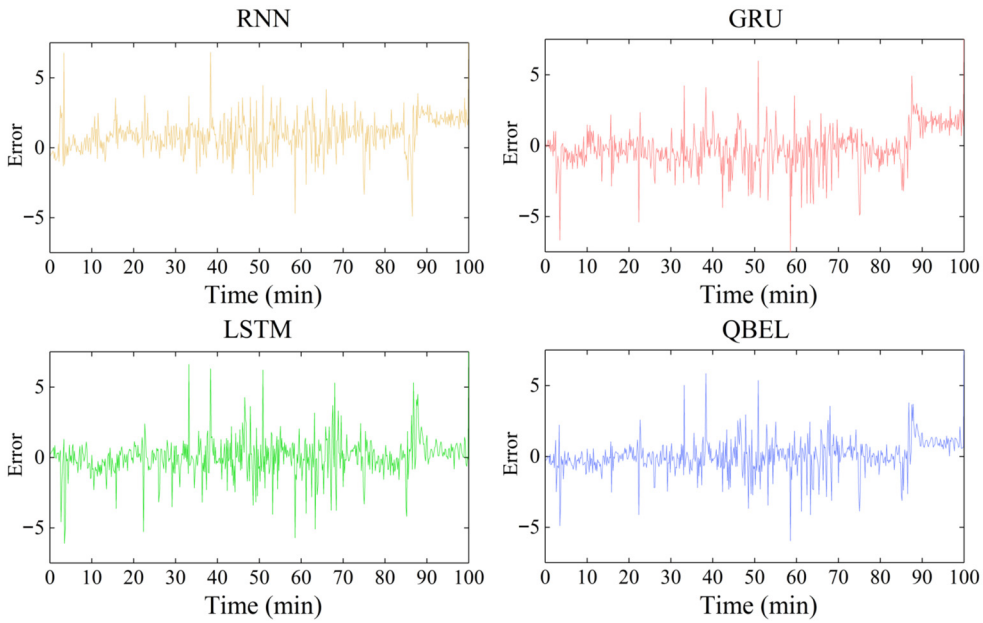

Figure 8. Prediction-error trajectories over the testing set for RNN, GRU, LSTM, and QBEL.

Furthermore, comparing the distribution characteristics of absolute errors in the predictions of various models, as shown in Figure 9, reveals differences in performance. The QBEL model outperforms other models. Specifically, the QBEL model exhibits the lowest median prediction error, indicating the slightest overall error among the models. Moreover, the QBEL model demonstrates the lowest variance in prediction errors, indicating superior stability and more reliable prediction outcomes compared to other models. Additionally, the performance of the LSTM model is slightly inferior to that of the QBEL model. In contrast, both the RNN and GRU models exhibit a wider distribution range of prediction errors, indicating higher uncertainty in their predictions.

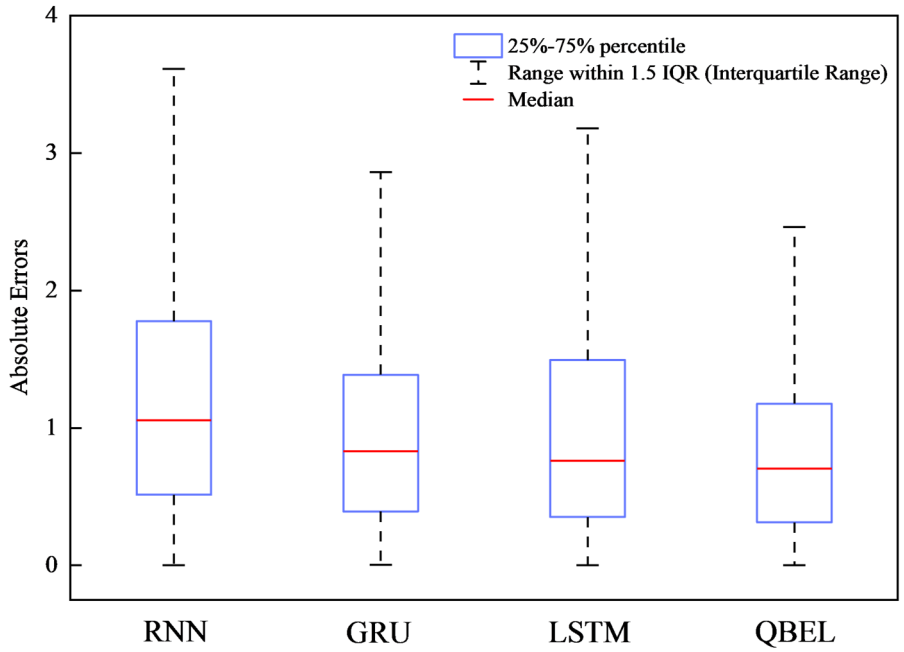

Figure 9. Absolute errors of prediction results for four models.

Figure 10 compares predicted and observed risk values on the testing set for QBEL and the three base learners. For each model, a least-squares line was fitted, and the coefficient of

determination (R^2) was reported. Agreement was therefore assessed jointly by three criteria: the fitted line's slope and intercept relative to the identity line (y = x), the value of R^2, and the proportion of points within the ±10% error band. QBEL achieves the highest R^2 (0.8420) and the largest within-band coverage (97.17%), whereas the other models displayed more dispersed points and lower coverage (RNN: 91.85%, GRU: 94.68%, LSTM: 95.67%). The tighter clustering of QBEL predictions around the identity line indicates reduced dispersion and smaller apparent bias, which is consistent with more reliable point forecasting under the same testing conditions.

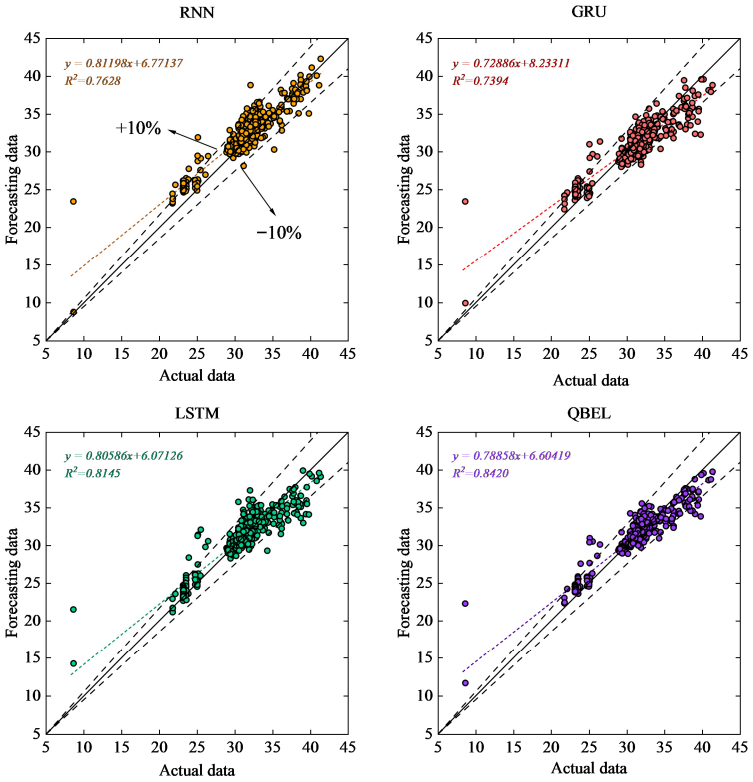

Figure 10. Predicted versus actual scatter plot on the testing set for QBEL and base learners. The black solid line denotes the equivalence between predicted values and actual values; The area enclosed by black dashed lines represents the error range where the predicted value deviates by ≤±10% from the actual value; Colored dashed lines are fitting lines for the overall data distribution. The closer a colored dashed line is to the black solid line, the better the overall prediction performance.

The improvement is attributable to the adaptive fusion mechanism. Fixed-weight averaging assumes that the relative skill of base learners is stable over time, which is often violated for nonstationary time series. Dynamic ensemble weighting and model selection have been widely reported to improve forecasting robustness when regimes shift, because the fusion rule can respond to performance feedback rather than relying on a single global combination [59,60]. The Q-learning-based weight update explicitly rewards loss reduction and therefore tends to suppress locally degraded base predictions, which explains the reduced spread and higher within-band coverage.

From an operational perspective, the ±10% band is a practical tolerance for alarm-oriented use, as predictions within this band are less likely to alter a risk-based decision when thresholds are applied. Forecast systems that generate frequent, large errors can increase false alarms or missed alarms, and degrade user trust in warning decisions. Therefore, the higher within-band coverage of QBEL supports more stable decision-making. Outliers persist for all models, particularly in regions with limited sample sizes or during rapid transitions, indicating that short-term disturbances, measurement noise, or unmodeled

factors may influence some events. These cases motivate further analysis of the residual structure near decision thresholds and, if necessary, the implementation of additional robustness measures in feature design or training loss to mitigate rare, large deviations.

4.2.3. QBEL vs. Other Time-Series Prediction Methods

Table 4 shows that DLinear achieved relatively stable and competitive accuracy (MSE = 1.6936), despite its small parameter scale, suggesting the utility of decomposition-based linear modeling for short-term risk forecasting. PatchTST demonstrated reasonable accuracy when the patch length was moderate (p = 8 or 16), but performance degraded substantially when p = 32, indicating that excessively long patches may obscure local temporal variations critical for risk prediction. MoLE exhibited non-monotonic behavior. Specifically, the best performance was observed at e = 8, whereas both smaller (e = 4) and larger (e = 16) configurations led to pronounced increases in error, implying challenges in balancing expert diversity and preventing overfitting. Fredformer consistently yielded the weakest results across all tested configurations (MSE > 4.7), reflecting its limited ability to accommodate non-stationary fluctuations inherent in fire risk dynamics. Collectively, these observations highlight the sensitivity of Transformer-based and mixture-of-experts models to hyperparameter tuning, as well as their lack of robustness in this application context. To be specific, QBEL attains the lowest MAE and MAPE among all competing methods, while its MSE is only marginally higher than that of the best linear baseline. In other words, compared with DLinear, QBEL reduces the average absolute error by around 10% and the mean absolute percentage error by roughly 33%, at the cost of only a minimal increase in squared error. Relative to the most competitive Transformer variant, QBEL still lowers MAE and MAPE by approximately one-tenth and about one-third, respectively. This pattern is consistent with the different sensitivities of the metrics: MSE/RMSE over-emphasize a few relatively large errors, whereas MAE and MAPE better reflect the typical error magnitude over most samples. The observed trade-off therefore suggests that QBEL provides more reliable predictions for most time steps, which is desirable for practical early-warning applications where stable accuracy is more critical than minimizing a few outlier errors.

Table 4. The comparison of QBEL and the other time-series forecasting models.

Model	Params	Accurate Metrics				Computational Metrics		
		MSE	RMSE	MAE	MAPE (%)	GFLOPs	Memory (MB)	Time (s)
Dlinear [53]	769	**1.6936**	**1.3014**	0.9599	3.9167	-	68.4095	2.8433
PatchTST (p = 8) [54]	569,025	1.9681	1.4029	0.9202	3.8311	0.0500	57.2063	0.4637
PatchTST (p = 16) [54]	556,465	1.9031	1.3795	1.0176	4.2262	0.0252	56.8316	0.8544
PatchTST (p = 32) [54]	565,953	3.0737	1.7532	1.2932	5.4703	0.0132	57.1133	0.6001
MoLE (e = 4) [55]	791,433	3.4377	1.8541	1.3398	5.6672	**0.0013**	60.5256	**0.0711**
MoLE (e = 8) [55]	1,580,433	2.2570	1.5023	1.0696	4.3897	0.0025	72.6062	0.1178
MoLE (e = 16) [55]	3,158,433	3.0038	1.7332	1.2415	5.2261	0.0031	96.7520	0.1800
Fredformer (b = 3, w = 2) [56]	628,609	4.7129	2.1709	1.7455	7.4094	0.2033	**35.0863**	0.4720
Fredformer (b = 2, w = 3) [56]	546,433	5.1701	2.2738	1.8306	7.7466	0.1368	35.4584	0.4541
Fredformer (b = 6, w = 1) [56]	1,071,745	5.0946	2.2571	1.7171	7.3420	0.4601	48.5279	0.1548
QBEL (this work)	552,192	1.7287	1.3148	**0.8352**	**2.6620**	0.0055	45.1514	0.0845

Note: The dash symbol ("-") denotes that the model is not deep learning-based, and GFLOPs is not applicable. The bolded values in this table represent the best performance for the corresponding metrics.

From a modeling perspective, DLinear, as a decomposition-based linear model, demonstrates that simple trend-seasonal separation can already yield competitive accuracy with very low complexity. However, its reliance on fixed linear structures prevents it from adapting to abrupt shifts or complex nonlinearities that characterize the evolution of cable fire risk, which is precisely reflected in its higher absolute and relative errors compared

with QBEL. To address such limitations, PatchTST introduces a Transformer backbone that captures long-range dependencies through patch embeddings, thereby complementing the short-sightedness of DLinear. Yet Table 4 indicates that its performance is susceptible to the choice of patch length: while moderate patch sizes deliver reasonable accuracy, enlarging the patch leads to increases of roughly 25–55% across error metrics, indicating that over-smoothing local fluctuations undermines robustness in practice. MoLE, with its mixture-of-experts design, offers an alternative perspective by enabling different experts to specialize in distinct dynamics, thereby attempting to balance the local-global trade-off observed in PatchTST. Nevertheless, its behavior is markedly non-monotonic across different numbers of experts: the best configuration reduces the errors of the poorest one by approximately 19–34% across MSE, RMSE, MAE, and MAPE, revealing a strong dependence on this hyperparameter and a lack of inherent adaptability in expert utilization. Fredformer shifts the inductive bias further by operating in the frequency domain, an appealing attempt to capture periodicities beyond the reach of time-domain models. However, its errors remain substantially larger than those of all other baselines, which are approximately 1.6–3.0 times those of the strongest competitors, depending on the metric, indicating that frequency-domain assumptions struggle with the irregular cycles and sudden non-stationary transitions present in fire-risk dynamics [61].

From a computational standpoint, QBEL also strikes a favorable balance between accuracy and efficiency. Its parameter count is of the same order as the Transformer baselines, which is markedly smaller than that of the most complex mixture-of-experts configuration. At the same time, its computational cost remains significantly lower than that of the heaviest architectures, such as Fredformer. In other words, QBEL achieves its gains in MAE and MAPE without incurring a proportional increase in complexity, yielding a more substantial accuracy-efficiency trade-off than both linear and nonlinear competitors. The combination of more stable performance and a better accuracy-cost balance is consistent with previous studies, which have shown that reinforcement-learning-based ensembles are less sensitive than static averaging or fixed meta-learners in time-series tasks [60,62,63].

4.3. Analysis of QBEL Model Results

4.3.1. Analysis of the Input to the Model

In many cases, univariate models are commonly employed for time series prediction, wherein future actual values are predicted based on the recorded values of the target variable. However, multivariate models offer the advantage of capturing interactions among multiple variables, thereby enhancing the accuracy of predictions. In this section, a comparative analysis is conducted to demonstrate the superiority of multivariate models over univariate models. Furthermore, the impact of varying input time step lengths on prediction results is examined to assess the influence of this crucial factor. The prediction errors of both multivariate and univariate models are evaluated under different input time step lengths, providing valuable insights into their performance. For the multivariate model, the input time series includes current, voltage, power, environmental temperature, humidity, oxygen content, and system risk value at each time step. In contrast, the univariate model only takes the system risk value at each time step as input. Taking LSTM as an example, the variations of MSE, RMSE, MAE, and MAPE for both univariate and multivariate models with respect to the input time step length are shown in Figure 11.

The trends in MSE and RMSE are consistent: for the univariate model, the prediction error reaches its minimum at an input time step length of 11. Below 11, a decreasing trend is observed, while above 11, an increasing trend is seen. As for the multivariate model, the prediction error reaches its minimum at an input time step length of 10. Below 10,

it generally decreases with increasing time step length; above 10, it exhibits an overall increasing trend.

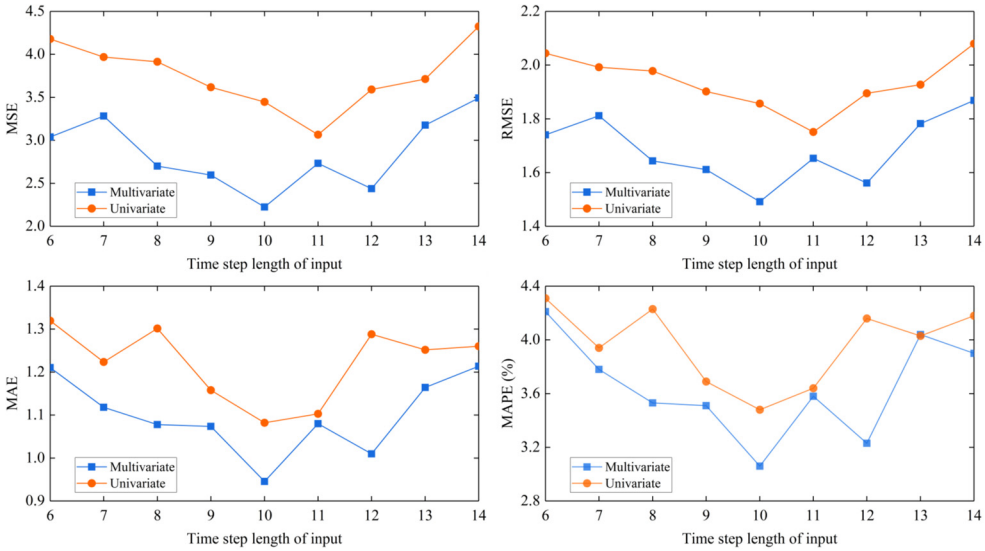

Figure 11. Metrics of univariate and multivariate models under different input time step lengths.

The trends in MAE and MAPE are also consistent: both the univariate and multivariate models achieve the minimum prediction error at an input time step length of 10. Below 10, the overall trend is a decrease with increasing time step length, and above 10, it shows an overall increasing trend.

A too-large input time step length (greater than 14) may prevent the models from capturing short-term fluctuations and trends, leading to less accurate predictions. In contrast, an input time step length that is too small (less than 6) allows the models to understand rapid changes and short-term trends better. However, it may limit their ability to capture longer-term patterns, potentially resulting in lower prediction accuracy. With the increase in input time step length, the errors show a constant fluctuation without a clear pattern. However, when using the same input time step length, the multivariate model consistently outperforms the univariate model regarding prediction accuracy. Therefore, the multivariate model demonstrates superior predictive performance based on the comparison.

4.3.2. Analysis of the Time Interval for Prediction

Predicting over an extended time interval allows for earlier fire risk warnings and provides more preparation time for proactive measures. In this section, to evaluate the practicability of the proposed model, the time interval for prediction was adjusted to forecast fire risk values for the subsequent states, 20 s and 30 s ahead, respectively. The model's input remains unchanged at the exact moment, with a feature dimension of 7 and a time step length of 10. Expanding the time interval for prediction involves adjusting the output to forecast the system's fire risk values further into the future. For a prediction time interval of 20 s, the input at time t remains X_t as in Equation (25), and the corresponding output y_t is R_{t+2}. For a prediction time interval of 30 s, the output y_t at time t corresponds to R_{t+3}.

The errors in the predicted results for each model are shown in Figure 12. Overall, it can be observed that the prediction accuracy of the model tends to decrease as the time interval for prediction lengthens. Furthermore, across the four metrics, the QBEL model exhibits the lowest prediction errors in both intervals, indicating that the proposed model achieves superior predictions across different forecast intervals.

The error improvement percentages for the two prediction intervals are presented in Table 5. It is evident that as the time interval for prediction increases, the relative

improvement of the QBEL model in error percentage over the single models increases, emphasizing a more pronounced reduction in errors.

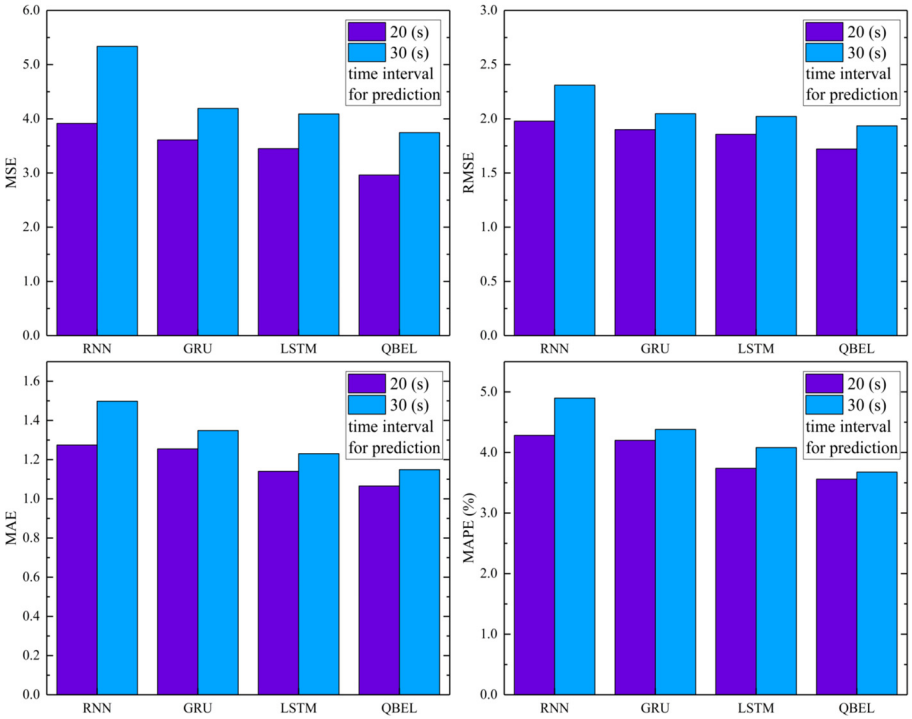

Figure 12. The error evaluation for different models in two types of intervals for prediction.

Table 5. The improvement of QBEL model over individual models across different time intervals.

Time Interval (s)	Model	P_{MSE} (%)	P_{RMSE} (%)	P_{MAE} (%)	P_{MAPE} (%)
20	RNN	24.3	13.0	16.4	16.9
	GRU	17.9	9.4	15.1	15.2
	LSTM	14.1	7.3	6.6	4.8
30	RNN	29.8	16.2	23.3	24.9
	GRU	10.7	5.5	14.8	16.1
	LSTM	8.4	4.3	6.6	9.9

4.3.3. Robustness Against Data Contamination

We conducted stress tests by injecting synthetic contamination into the base learners' outputs on the test set, aiming to emulate the downstream effect of sensor noise, intermittent communication loss, and abnormal operational disturbances on model inference.

Noise injection and sensor dropout are two contamination mechanisms were considered. As for noise injection, a fraction ρ of time points was randomly selected and perturbed by additive Gaussian noise with standard deviation $\sigma = \alpha \cdot std(\hat{y})$. As for sensor dropout, a fraction ρ of time points was randomly dropped (set to NaN) and then imputed using a simple forward-fill strategy to mimic temporary missing streams. For each $\rho \in \{0, 0.05, 0.10, 0.20, 0.30, 0.40, 0.50\}$, we repeated the perturbation and evaluation 50 times (Monte Carlo) and reported the mean RMSE. Figure 13 illustrates the results for noise injection (a) and dropout (b).

In the clean condition ($\rho = 0$), the reported baseline RMSEs are 1.678 (RNN), 1.602 (LSTM), and 1.491 (GRU), while the best-performing QBEL setting achieves 1.315 RMSE, corresponding to an 11.841% reduction relative to the best single learner (GRU). Under contamination, QBEL remains consistently more stable. For example, under noise injection at $\rho = 0.100$, the best base learner reaches 6.061 RMSE while QBEL yields

4.098 RMSE (32.388% lower). At $\rho = 0.300$, the best base learner reaches 9.983 RMSE and QBEL yields 6.866 RMSE (31.228% lower). Under dropout, at $\rho = 0.100$, the best base learner achieves 4.662 RMSE while QBEL achieves 3.481 RMSE (25.325% lower); at $\rho = 0.300$, the best base learner reaches 8.061 RMSE while QBEL reaches 6.634 RMSE (17.697% lower).

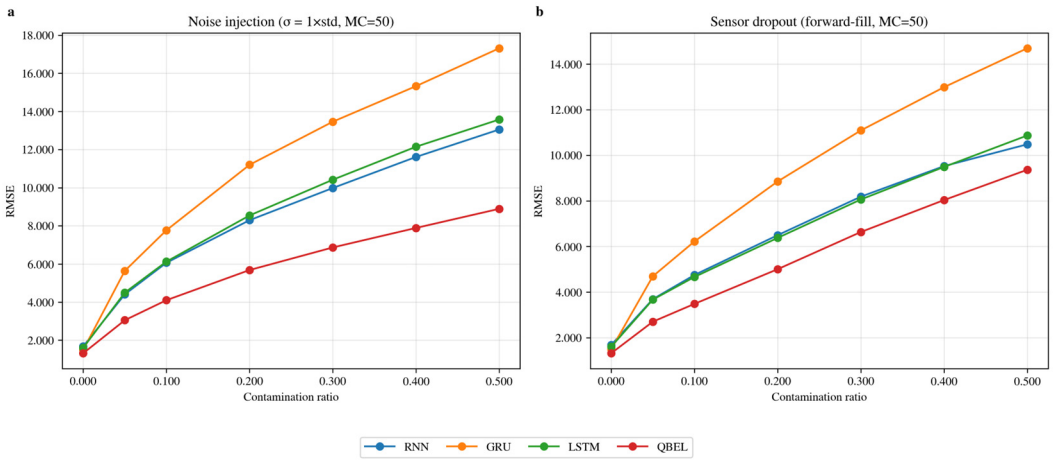

Figure 13. Robustness comparison under synthetic contamination of base-learner predictions: (**a**) noise injection ($\sigma = 1 \cdot std$); (**b**) sensor dropout with forward-fill imputation. Curves show the mean RMSE over 50 Monte-Carlo runs across the contamination ratio ρ.

4.4. Discussions

The multivariate prediction models implemented using RNN, LSTM, and GRU exhibit sound predictive performance, indicating the potential of deep structures in comprehensively recognizing the fluctuation characteristics of multivariate time series.

The multivariate advantage over univariate formulations is plausibly explained by the ability to capture nonlinear cross-feature dependencies and leverage complementary indicators across electrical and environmental channels. The dependence on window length reflects a context–noise trade-off: very short windows can limit temporal context, whereas overly long windows can introduce irrelevant history and amplify noise sensitivity. Under controlled stress testing that emulates degraded data quality, QBEL tends to degrade more gradually than the strongest single learner, which indicates that adaptive ensembles can reduce the impact of localized failures in individual predictors by reallocating weights.

Because QBEL aggregation reduces to a lightweight Q-table lookup and a weighted combination over three compact recurrent predictors, the added inference overhead remains modest. This computational profile suggests that QBEL can be integrated into edge-oriented monitoring units as an on-site forecasting module, operating in conjunction with existing electrical monitoring and alarm logic to enhance predictive accuracy while maintaining robustness against degraded data quality. In a practical workflow, sensor streams are first acquired and preprocessed at the edge for time alignment, filtering, normalization, and handling of missing data. The base learners then generate short-horizon risk predictions, which are fused by QBEL to produce the final risk forecast. This forecast can be mapped to warning outputs using conservative decision rules, while event logs are retained for audit and maintenance, supporting timely alerts without displacing conventional alarm safeguards.

5. Limitation and Future Work

The case study is constrained by the available data, which represent routine operation from one logistics facility over one day and a limited range of cable configurations. The re-

sults should therefore be interpreted as setting-specific evidence rather than a broad assessment across seasons, facilities, operating regimes, and cable types. The robustness analysis relies on controlled perturbations that approximate degraded data quality; however, real monitoring systems can exhibit more complex issues, such as correlated missingness, drift, and timing irregularities. In addition, the current study focuses on algorithmic evaluation rather than a field pilot integrated with operational fire alarm or electrical monitoring hardware, and deployment-facing validation remains to be established. Future work will therefore concentrate on expanding multi-site and longer-duration datasets with more diverse cables, evaluating the full pipeline under deployment-representative disturbances through hardware-in-the-loop testing, and extending forecasting to longer horizons with reliable outputs. Additionally, future work should conduct a deployment-focused pilot, implementing the end-to-end pipeline on an edge or gateway device and integrating it into alarm workflows, to further establish practical deployment capabilities and enable a comprehensive evaluation of latency, alarm stability, and false alarm control.

6. Conclusions

Accurate prediction of cable fire risk is important for preventing and managing complex environmental fires. This study addresses the limitations of conventional fire detection approaches, including delayed alerts and limited predictive fidelity, by proposing a QBEL model for predicting cable fire risk. QBEL integrates three deep predictors, RNN, LSTM, and GRU, utilizing a tabular Q-learning framework that adaptively allocates ensemble weights based on recent prediction errors and evolving risk patterns.

In the presented case study, QBEL achieves improved predictive performance, reducing MSE by 22.3 to 38.6% compared with individual base learners. This improvement is attributed to two design elements: a Q-learning-based weighting mechanism that can adapt to locally non-stationary risk transitions, and the use of multivariate inputs, such as current, voltage, and temperature, to represent interactions among risk-related factors. The experiments also demonstrate consistent benefits of multivariate modeling over univariate baselines, with the best performance achieved at an input time step length of 10 s.

QBEL further exhibits competitive performance relative to conventional ensemble strategies and recent time-series forecasting models. Compared to boosting and stacking ensembles built on the same base learners, QBEL achieves comparable or better accuracy while exhibiting lower sensitivity to hyperparameter settings and a moderate computational cost. When compared with DLinear, PatchTST, MoLE, and Fredformer, QBEL achieves the lowest MAE and MAPE among the evaluated methods, while maintaining MSE close to the best linear baseline and requiring fewer parameters and less memory than the most complex architectures.

Overall, this work presents a data-driven framework that combines deep learning with reinforcement learning to support cable fire-risk forecasting in building monitoring contexts. By capturing both short-term and long-term dependencies and adapting to changing risk patterns, QBEL provides a complementary tool for enhancing early-stage hazard identification in electrical subsystems. Future work will expand data coverage across various cable types and operating conditions, and conduct deployment-oriented pilot studies that implement the pipeline on an edge or gateway device, interfacing it with practical alarm workflows to enhance the timeliness and reliability of risk alerts.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/fire9020061/s1, Figure S1: Fire risk evaluation algorithm; Figure S2: Wire temperature rise experiment; Figure S3: Cable temperature rise experiment and the effect of ambient temperature on aging acceleration factor; Figure S4: Cable burning experiment with different oxygen content and the burning acceleration effect; Figure S5: Statistical distribution of RH weights

with respect to fire risk under different current ranges; Figure S6: Cable fire risk value variation under ±5% and ±10% perturbations of individual variables, reported as median ΔR_{fire} with the 5th to 95th percentile interval; Figure S7: Risk value response curves with single-variable sweeps; Figure S8: Correlation structure of operating variables and Mantel-test associations with the cable fire risk value; Table S1: Weights of RH across different current ranges as determined by 7 experts. File S1: Experts' scores on risk factors. Supplementary References: [64–66].

Author Contributions: Conceptualization, H.L.; Methodology, H.L.; Software, H.G.; Validation, X.G.; Resources, X.G. and G.H.; Data curation, H.G.; Writing—original draft, H.L.; Visualization, H.G.; Supervision, X.G.; Project administration, G.H.; Funding acquisition, G.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Key Research and Development Program of China with Grant No. 2023YFC3009805.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: Author Hao Gao was employed by the company Beijing BOE Chuang Yuan Technology Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Berg, H.; Piljugin, E.; Herb, J.; Röwekamp, M. Comprehensive cable failures analysis for probabilistic fire safety assessments. *Reliab. Theory Appl.* **2012**, *7*, 36–49.
2. Huang, X.; Nakamura, Y. A Review of Fundamental Combustion Phenomena in Wire Fires. *Fire Technol.* **2019**, *56*, 315–360. [CrossRef]
3. Marti, J.; Schartel, B.; Oate, E. Simulation of the burning and dripping cables in fire using the particle finite element method. *J. Fire Sci.* **2022**, *40*, 3–25. [CrossRef]
4. Park, J.H.; Lee, S.; Yun, S.; Kim, H.; Kim, W.T. Dependable Fire Detection System with Multifunctional Artificial Intelligence Framework. *Sensors* **2019**, *19*, 2025. [CrossRef] [PubMed]
5. Li, Z.B.; Hong, Z. Research on the application of fuzzy data fusion to cable fire detecting system. In Proceedings of the 2004 International Conference on Machine Learning and Cybernetics, Shanghai, China, 26–29 August 2004.
6. Yang, K.; Zhang, R.; Yang, J.; Liu, C.; Chen, S.; Zhang, F. A Novel Arc Fault Detector for Early Detection of Electrical Fires. *Sensors* **2016**, *16*, 500. [CrossRef] [PubMed]
7. Ferreira, T.M. Current Advances on the Assessment and Mitigation of Fire Risk in Buildings and Urban Areas—First Edition. *Fire* **2023**, *6*, 454. [CrossRef]
8. Kumar, K.; Sundar, G.; Thakur, M. Advanced Applications of Neural Networks and Artificial Intelligence: A Review. *Int. J. Inf. Technol. Comput. Sci.* **2012**, *4*, 57–68. [CrossRef]
9. Bu, F.; Gharajeh, M.S. Intelligent and vision-based fire detection systems: A survey. *Image Vis. Comput.* **2019**, *91*, 103803. [CrossRef]
10. Deb, C.; Zhang, F.; Yang, J.; Lee, S.E.; Shah, K.W. A review on time series forecasting techniques for building energy consumption. *Renew. Sustain. Energy Rev.* **2017**, *74*, 902–924. [CrossRef]
11. Chen, X.; Huang, G.; Gao, X.; Ou, S.; Li, Y.; Hezam, I.M. BN-RA: A Hybrid Model for Risk Analysis of Overload-Induced Early Cable Fires. *Appl. Sci.* **2021**, *11*, 8922. [CrossRef]
12. Zhu, H.; Ji, J.; Nie, J. Early fire evolution and alarm characteristics of cable fires in long and narrow spaces. *Fire Saf. J.* **2022**, *131*, 103627. [CrossRef]
13. Liu, X.; Hou, D.; Ji, J.; Zhu, H. Experiment and numerical simulation of cable trench fire detection. *Case Stud. Therm. Eng.* **2021**, *28*, 101338. [CrossRef]
14. Wang, Y.; Zhang, X.; Wang, L.; Wang, Y. Detection and Prediction of Internal-Caused Fire in Tunnel Cable by an Equivalent Transient Thermal Circuit Model. *Adv. Civ. Eng.* **2021**, *2021*, 5618575. [CrossRef]
15. Han, J.; Chen, W.; Yu, A.; Yi, J. Detection of Semi-volatile Plasticizers as a Signature of Early Electrical Fire. *Front. Mater.* **2019**, *6*, 250. [CrossRef]
16. Sun, K.; Zhao, Q.; Wang, X. Using knowledge inference to suppress the lamp disturbance for fire detection. *J. Saf. Sci. Resil.* **2021**, *2*, 124–130. [CrossRef]
17. Xiao, S.; Wang, S.; Ge, L.; Weng, H.; Fang, X.; Peng, Z.; Zeng, W. Hybrid Feature Fusion-Based High-Sensitivity Fire Detection and Early Warning for Intelligent Building Systems. *Sensors* **2023**, *23*, 859. [CrossRef] [PubMed]

18. Li, C.; Chen, J.; Hu, L.; Zhang, W.; Cao, J.; Tan, X.; Li, H. Research on Cable Fire Detection and Early Warning Based on Deep Learning. In Proceedings of the 2021 IEEE International Conference on Electronic Technology, Communication and Information (ICETCI), Changchun, China, 27–29 August 2021; pp. 176–179.
19. Sheng, D.; Deng, J.; Xiang, J. Automatic Smoke Detection Based on SLIC-DBSCAN Enhanced Convolutional Neural Network. *IEEE Access* **2021**, *9*, 63933–63942. [CrossRef]
20. Diaconu, B.M. Recent Advances and Emerging Directions in Fire Detection Systems Based on Machine Learning Algorithms. *Fire* **2023**, *6*, 441. [CrossRef]
21. Jin, C.; Wang, T.; Alhusaini, N.; Zhao, S.; Liu, H.; Xu, K.; Zhang, J. Video Fire Detection Methods Based on Deep Learning: Datasets, Methods, and Future Directions. *Fire* **2023**, *6*, 315. [CrossRef]
22. Vuppari, G.R.; Gupta, N.; El-Sayed, A.; Xiong, X. Wildfire Detection Using Vision Transformer with the Wildfire Dataset. *arXiv* **2025**, arXiv:2505.17395. [CrossRef]
23. Ahmad, N.; Akbar, M.; Alkhammash, E.H.; Jamjoom, M.M. CN2VF-Net: A Hybrid Convolutional Neural Network and Vision Transformer Framework for Multi-Scale Fire Detection in Complex Environments. *Fire* **2025**, *8*, 211. [CrossRef]
24. Zhu, J.; Zhang, J.; Wang, Y.; Ge, Y.; Zhang, Z.; Zhang, S. Fire Detection in Ship Engine Rooms Based on Deep Learning. *Sensors* **2023**, *23*, 6552. [CrossRef] [PubMed]
25. Du, S.; Li, J.; Noto, M. Comparison and analysis of three mobilenet-based models for wildfire detection. *J. Adv. Inf. Technol.* **2024**, *15*, 511–518. [CrossRef]
26. Huang, J.; Yang, H.; Liu, Y.; Liu, H. A Forest Fire Smoke Monitoring System Based on a Lightweight Neural Network for Edge Devices. *Forests* **2024**, *15*, 1092. [CrossRef]
27. Nadeem, M.; Dilshad, N.; Alghamdi, N.S.; Dang, L.M.; Song, H.-K.; Nam, J.; Moon, H. Visual Intelligence in Smart Cities: A Lightweight Deep Learning Model for Fire Detection in an IoT Environment. *Smart Cities* **2023**, *6*, 2245–2259. [CrossRef]
28. Wei, C.; Xu, J.; Li, Q.; Jiang, S. An Intelligent Wildfire Detection Approach through Cameras Based on Deep Learning. *Sustainability* **2022**, *14*, 15690. [CrossRef]
29. Jin, L.; Yu, Y.; Zhou, J.; Bai, D.; Lin, H.; Zhou, H. SWVR: A Lightweight Deep Learning Algorithm for Forest Fire Detection and Recognition. *Forests* **2024**, *15*, 204. [CrossRef]
30. Han, Z.; Zhao, J.; Leung, H.; Ma, K.F.; Wang, W. A Review of Deep Learning Models for Time Series Prediction. *IEEE Sens. J.* **2021**, *21*, 7833–7848. [CrossRef]
31. Zhou, X.; Wang, J.; Liu, Y.; Duan, Q. Deep learning with PID residual elimination network for time-series prediction of water quality in aquaculture industry. *Comput. Electron. Agric.* **2023**, *212*, 108125. [CrossRef]
32. Yan, J.; Zhang, Y.; Su, Q.; Li, R.; Li, H.; Lu, Z.; Lu, H.; Lu, Q. Time series prediction based on LSTM neural network for top tension response of umbilical cables. *Mar. Struct.* **2023**, *91*, 103448. [CrossRef]
33. Shen, G.; Tan, Q.; Zhang, H.; Zeng, P.; Xu, J. Deep Learning with Gated Recurrent Unit Networks for Financial Sequence Predictions. *Procedia Comput. Sci.* **2018**, *131*, 895–903. [CrossRef]
34. Li, S.; Huang, H.; Lu, W. A Neural Networks Based Method for Multivariate Time-Series Forecasting. *IEEE Access* **2021**, *9*, 63915–63924. [CrossRef]
35. Arikan, O.; Uydur, C.C.; Kumru, C.F. Prediction of dielectric parameters of an aged mv cable: A comparison of curve fitting, decision tree and artificial neural network methods. *Electr. Power Syst. Res.* **2022**, *208*, 107892. [CrossRef]
36. Ledari, S.A.; Mirzaie, M. Sheath induced voltage prediction of high voltage cable based on artificial neural network. *Comput. Electr. Eng.* **2020**, *87*, 106788. [CrossRef]
37. Thanh, P.N.; Yeh, C.-T.; Cho, M.-Y. Predicting Leakage Current of Distribution Insulators Based Deep Learning Gated Recurrent Unit. In Proceedings of the 2022 5th Asia Conference on Energy and Electrical Engineering (ACEEE), Kuala Lumpur, Malaysia, 8–10 July 2022; pp. 46–50.
38. Peng, J.; Yu, W. The algorithm of current prediction based on multi-dimensional Long Short Term Memory networks. *Energy Rep.* **2021**, *7*, 1114–1120. [CrossRef]
39. Pourhosseini, F.A.; Ebrahimi, K.; Omid, M.H. Prediction of total dissolved solids, based on optimization of new hybrid SVM models. *Eng. Appl. Artif. Intell.* **2023**, *126*, 106780. [CrossRef]
40. Wang, H.-K.; Song, K.; Cheng, Y. A Hybrid Forecasting Model Based on CNN and Informer for Short-Term Wind Power. *Front. Energy Res.* **2022**, *9*, 788320. [CrossRef]
41. Bai, Y.; Chen, Z.; Xie, J.; Li, C. Daily reservoir inflow forecasting using multiscale deep feature learning with hybrid models. *J. Hydrol.* **2016**, *532*, 193–206. [CrossRef]
42. Liu, H.; Mi, X.-W.; Li, Y.-F. Wind speed forecasting method based on deep learning strategy using empirical wavelet transform, long short term memory neural network and Elman neural network. *Energy Convers. Manag.* **2018**, *156*, 498–514. [CrossRef]
43. Sattarzadeh, A.R.; Kutadinata, R.J.; Pathirana, P.N.; Huynh, V.T. A novel hybrid deep learning model with ARIMA Conv-LSTM networks and shuffle attention layer for short-term traffic flow prediction. *Transp. A Transp. Sci.* **2023**, *21*, 2236724. [CrossRef]

44. Fu, Y.; Wu, D.; Boulet, B. Reinforcement Learning Based Dynamic Model Combination for Time Series Forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtually, 22 February–1 March 2022; Volume 36, pp. 6639–6647.
45. Li, Y.; Liu, Z.; Liu, H. A novel ensemble reinforcement learning gated unit model for daily PM2.5 forecasting. *Air Qual. Atmos. Health* **2020**, *14*, 443–453. [CrossRef]
46. Tan, J.; Liu, H.; Li, Y.; Yin, S.; Yu, C. A new ensemble spatio-temporal PM2.5 prediction method based on graph attention recursive networks and reinforcement learning. *Chaos Solitons Fractals* **2022**, *162*, 112405. [CrossRef]
47. Juliá, P.B.; Ferreira, T.M.; Luco, C.A.; Sandoval, C.; Palazzi, N.C.; Hurtado, M.; Oliveira, D.V. Combining Physical and Social Indicators in a Multi-Vulnerability Framework for Fire Risk Assessment: An application to Valparaíso, Chile. *Int. J. Disaster Risk Reduct.* **2025**, *131*, 105900. [CrossRef]
48. Ferreira, T.M.; Vicente, R.; da Silva, J.A.R.M.; Varum, H.; Costa, A.; Maio, R. Urban fire risk: Evaluation and emergency planning. *J. Cult. Herit.* **2016**, *20*, 739–745. [CrossRef]
49. Hopfield, J.J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA* **1982**, *79*, 2554–2558. [CrossRef]
50. Elman, J.L. Finding structure in time. *Cogn. Sci.* **1990**, *14*, 179–211. [CrossRef]
51. Yeh, W.C. Application of Long Short-Term Memory Recurrent Neural Networks Based on the BAT-MCS for Binary-State Network Approximated Time-Dependent Reliability Problems. *arXiv* **2022**, arXiv:2202.07837.
52. Guo, Z.; Yang, C.; Wang, D.; Liu, H. A novel deep learning model integrating CNN and GRU to predict particulate matter concentrations. *Process Saf. Environ. Prot.* **2023**, *173*, 604–613. [CrossRef]
53. Zeng, A.; Chen, M.; Zhang, L.; Xu, Q. Are transformers effective for time series forecasting? In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; pp. 11121–11128.
54. Nie, Y.; Nguyen, N.H.; Sinthong, P.; Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv* **2022**, arXiv:2211.14730.
55. Ni, R.; Lin, Z.; Wang, S.; Fanti, G. Mixture-of-linear-experts for long-term time series forecasting. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Valencia, Spain, 2–4 May 2024; pp. 4672–4680.
56. Piao, X.; Chen, Z.; Murayama, T.; Matsubara, Y.; Sakurai, Y. Fredformer: Frequency Debiased Transformer for Time Series Forecasting. In Proceedings of the Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Barcelona, Spain, 25–29 August 2024; pp. 2400–2410.
57. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
58. Koop, G.; Korobilis, D. Forecasting inflation using dynamic model averaging. *Int. Econ. Rev.* **2012**, *53*, 867–886. [CrossRef]
59. de Carvalho, H.D.P.; de Oliveira, J.F.L.; Fagundes, R.A.d.A. Dynamic selection of ensemble-based regression models: Systematic literature review. *Expert Syst. Appl.* **2025**, *290*, 128429. [CrossRef]
60. Ding, Y.; Park, Y.; Gopalswamy, K.; Hasson, H.; Wang, B.; Huan, L. Dynamic ensemble for probabilistic time-series forecasting via deep reinforcement learning. In Proceedings of the Eleventh International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
61. Liu, Y.; Wu, H.; Wang, J.; Long, M. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 9881–9893.
62. Wu, Z.; Fang, G.; Ye, J.; Zhu, D.Z.; Huang, X. A reinforcement learning-based ensemble forecasting framework for renewable energy forecasting. *Renew. Energy* **2025**, *244*, 122692. [CrossRef]
63. He, X.; Zhao, W.; Zhang, L.; Zhang, Q.; Li, X. A novel ensemble deep reinforcement learning model for short-term load forecasting based on Q-learning dynamic model selection. *J. Eng.* **2024**, *2024*, e12409. [CrossRef]
64. Riba, J.-R.; Llauradó, J. A model to calculate the current–temperature relationship of insulated and jacketed cables. *Materials* **2022**, *15*, 6814. [CrossRef] [PubMed]
65. Yang, Z.; Yao, W.; Zhang, J.; Li, C.; Bo, B.; Lu, S. Combustion characteristics and fire hazard assessment of small-scale pool fires in increased oxygen concentration. *Fire Saf. J.* **2024**, *144*, 104114. [CrossRef]
66. Marquis, D.; Guillaume, E.; Camillo, A. Effects of oxygen availability on the combustion behaviour of materials in a controlled atmosphere cone calorimeter. *Fire Saf. Sci.* **2014**, *11*, 138–151. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.