BMC Public Health

RESEARCH

Open Access

Forecasting and analyzing influenza activity in Hebei Province, China, using a CNN-LSTM hybrid model

Guofan Li[1†], Yan Li[2†], Guangyue Han[2], Caixiao Jiang[2], Minghao Geng[2], Nana Guo[2], Wentao Wu[2], Shangze Liu[2], Zhihuai Xing[2], Xu Han[2] and Qi Li[1,2*]

Abstract

Background Influenza, an acute infectious respiratory disease, presents a significant global health challenge. Accurate prediction of influenza activity is crucial for reducing its impact. Therefore, this study seeks to develop a hybrid Convolution Neural Network—Long Short Term Memory neural network (CNN-LSTM) model to forecast the percentage of influenza-like-illness (ILI) rate in Hebei Province, China. The aim is to provide more precise guidance for influenza prevention and control measures.

Methods Using ILI% data from 28 national sentinel hospitals in the Hebei Province, spanning from 2010 to 2022, we employed the Python deep learning framework PyTorch to develop the CNN-LSTM model. Additionally, we utilized R and Python to develop four other models commonly used for predicting infectious diseases. After constructing the models, we employed these models to make retrospective predictions, and compared each model's prediction performance using mean absolute error (MAE), root mean square error (RMSE), mean absolute percentage error (MAPE), and other evaluation metrics.

Results Based on historical ILI% data from 28 national sentinel hospitals in Hebei Province, the Seasonal Auto-Regressive Indagate Moving Average (SARIMA), Extreme Gradient Boosting (XGBoost), Convolution Neural Network (CNN), Long Short Term Memory neural network (LSTM) models were constructed. On the testing set, all models effectively predicted the ILI% trends. Subsequently, these models were used to forecast over different time spans. Across various forecasting periods, the CNN-LSTM model demonstrated the best predictive performance, followed by the XGBoost model, LSTM model, CNN model, and SARIMA model, which exhibited the least favorable performance.

Conclusion The hybrid CNN-LSTM model had better prediction performances than the SARIMA model, CNN model, LSTM model, and XGBoost model. This hybrid model could provide more accurate influenza activity projections in the Hebei Province.

Keywords Influenza, Forecast, Deep Learning, Hybrid Model, Modeling

†Guofan Li and Yan Li contributed equally to this work.

*Correspondence:
Qi Li
liqinew@126.com
[1] School of Public Health, Hebei Medical University, No.361, Zhongshan East Road, Shijiazhuang, Hebei Province 050017, China
[2] Hebei Provincial Center for Disease Control and Prevention, No.97, Huai'an East Road, Shijiazhuang, Hebei Province 050021, China

Background

Influenza (also referred to as the flu) is an acute infectious respiratory disease. It is caused by two major influenza virus types, flu A and flu B. It is highly contagious, with a short incubation period and severe symptoms [1], and has a history of causing global pandemics, such as the Spanish flu in 1918, the Asian flu

in 1957, and the swine flu in 2009 [2]. According to the World Health Organization (WHO), there are approximately one billion cases of flu reported worldwide each year. Among these cases, 3 to 5 million are classified as severe, with 290,000 to 650,000 resulting in fatalities. Thus, flu seriously endangers individual and public health and is responsible for a great deal of economic burdens [3]. Flu typically manifests as an annual seasonal epidemic or a sporadic pandemic [4, 5], and understanding its activity patterns is crucial for effectively carrying out prevention and control measures. Currently, the primary measure against the flu epidemic is widespread vaccination. However, due to the variability of the flu virus and unexpected outbreaks, vaccines have effectiveness as low as 20% [6, 7]. As a result, analyzing historical surveillance data and making prediction about flu is of great importance.

Flu was the first infectious to be monitored by the WHO. The indicators monitored include influenza-like-illness (ILI), flu specimen positivity rates, and reported flu cases. The sensitivity of various indicators varies across regions. In southern China, the positivity rate of flu specimens serves as a better reflection of the flu epidemic situation, whereas in the north, the ILI proves to be more sensitive [8]. Conversely, the reported number of flu cases often lags significantly behind [9].

Flu surveillance data can be regarded as time series data. Constructing prediction models using previous data can allow for the creation of public health guidance for future epidemics. Traditional prediction models like logistic regression models, Extreme Gradient Boosting (XGBoost) models, and Seasonal Auto-Regressive Indagate Moving Average (SARIMA) models, have been utilized for predicting various contagious diseases such as hand, foot, and mouth disease [10], tuberculosis [11], and COVID-19 [12], and have demonstrated good prediction performance. Additionally, various deep learning methods including the Recursive Neural Network (RNN), the Convolution Neural Network (CNN), and the RNN-based improved Long Short Term Memory neural network (LSTM) have been applied across numerous domains. For instance, CNNs are used to detect anomalies in circuit signals [13], while LSTMs are utilized for predicting stock prices [14], weather patterns [15], and the number of cases [16]. In comparison to traditional prediction models, these methods have stronger nonlinear mapping and sequence information extraction capabilities [17]. In addition, different neural networks possess varying information processing capabilities; CNNs excel in information extraction, while LSTMs are more suitable for processing long sequence prediction tasks. Combining the strengths of these different models may enhance prediction performance.

Hebei Province, which is situated in northern China, typically experiences concentrated influenza activity during the winter and spring seasons [18]. In recent years, the number of reported flu cases during the flu season in Hebei Province has been increasing [19]. It is crucial to utilize surveillance data for predicting peak outbreaks in advance to effectively prevent and control flu in the region. However, there are few studies on flu prediction models, with most focusing on several sentinel hospitals in urban areas. Furthermore, these studies predominantly rely on traditional prediction models such as ARIMA and multiple regression models [20, 21]. However, compared to machine learning and deep learning models in other studies, their predictive results are not very good. Thus, there is a significant gap in the development of accurate and representative flu prediction models specifically tailored for Hebei Province. Here, we gathered ILI data reported by 28 sentinel hospitals across 11 cities in Hebei Province. We then developed a hybrid CNN-LSTM model (which leveraged the distinct advantages of both the CNN and LSTM architectures) on the PyTorch platform. Finally, due to the SARIMA and XGBoost models are frequently employed in time series forecasting and characterized by high accuracy. We chose the SARIMA model and the XGBoost model as the baseline models, and compared the predictive efficacy of our hybrid model to the SARIMA, XGBoost, CNN, and LSTM models using retrospective forecasting.

Methods

Figure 1 is a diagrammatic representation of our research methodology, including the materials and methods used to develop the SARIMA, XGBoost, CNN, LSTM, and CNN-LSTM models Forecasting Model for ILI fluctuation in the Hebei Province of China.

Data acquisition and splitting

Data were retrieved from the China Influenza Surveillance Information System, which gathers weekly reported ILI cases and related outpatient/emergency department visits from 28 national sentinel hospitals in the Hebei Province. Our data covered the period from the first week of 2010 to the 52nd week of 2022. Due to variations in the sizes of sentinel hospitals and the population bases of different areas, calculating the ILI percentage helps to minimize these factors [22]. ILI% were calculated based on ILI cases and visit data using the following formula:

$$ILI\% = \frac{numberofILIs}{totalnumberofoutpatient/emergencyvisits} \times 100\%.$$

We then split the dataset using an 80–20 ratio, with the initial 80% used as a training set and the remaining 20% used as a testing set for predicted results.

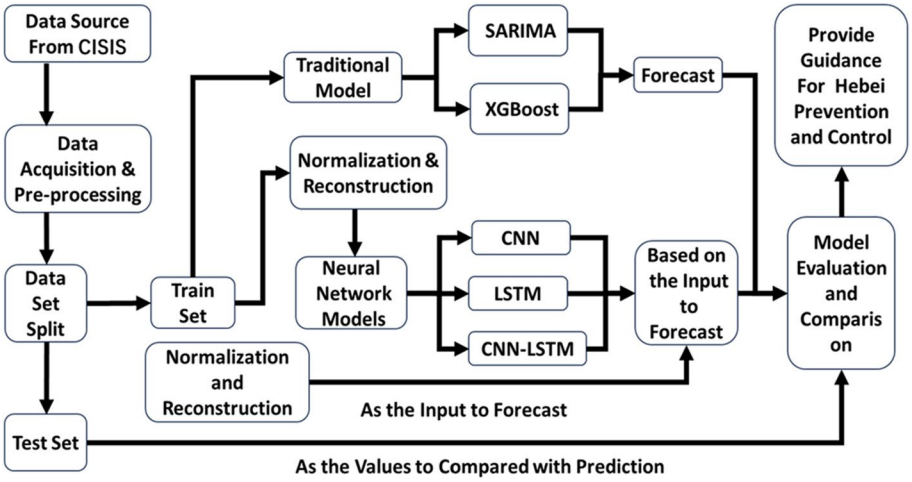

Fig. 1 Study methodology

Traditional models

SARIMA model

SARIMA models consist of four components: autoregression (AR), integration (I), moving average (MA), and seasonal effects (S) [23]. SARIMAs can be classified as either multiplicative or additive models, depending on the relationship between seasonal effects and other components. In a multiplicative model, the expression is SARIMA $(p,d,q)(P,D,Q)_s$, where p and P represent the orders of ordinary AR and seasonal AR, respectively; d and D represent the orders and the steps and orders of the difference function; q and Q represent the ordinary orders of MA and seasonal MA individually; and S represents the seasonal period. The SARIMA calculation formula was as follows [24]:

$$\begin{gathered}\phi_p(B)\Phi_P(B^S)(1-B)^d(1-B^S)^D y_t = \theta_q(B)\Theta_Q(B^S)u_t,\\ \phi(B) = 1-\phi_1(B)-\phi_2(B)^2-\ldots\ldots-\phi_p(B)^p,\\ \Phi\left(B^S\right) = 1-\Phi_1(B)^S-\Phi_2(B)^{2S}-\ldots\ldots-\Phi_P(B)^{PS},\\ \theta(B) = 1-\theta(B)-\theta_2(B)^2-\ldots\ldots-\theta_q(B)^q,\\ \Theta(B^S) = 1-\Theta(B)^S-\Theta_2(B)^{2S}-\ldots\ldots-\Theta_Q(B)^{QS},\\ E(u_t)=0, Var(u_t)=\sigma^2,\\ E(\mathrm{u_t},\mathrm{u_S})=0, S\neq t, E\left(y_s,u_t\right)=0, \forall_S < t_\circ\end{gathered}$$

In this formula, B is the lagging operator, ϕ and θ are the ordinary AR and MA coefficients, Φ and Θ are the seasonal AR and MA coefficients, s is the seasonal period, y_t is a zero-mean steady random time series, and u_t is a white noise sequence.

The SARIMA modeling process involves several key steps includes time series preprocessing, stationarity testing, parameter identification, and model fitting [25]. The initial data processing step includes conducting an augmented Dickey-Fuller (ADF) test and a white noise sequence test (Box-Ljung test) to determine whether the data represent a stationary and non-white noise sequence. If the data are non-stationary, differencing is necessary to achieve stationarity. The values of d and D are defined by the number and order of the differencing steps.

Next is the model identification phase, which involves examining the autocorrelation function (ACF) and partial autocorrelation function (PACF). Within these plots, the AR can be identified in a significant autocorrelation at lag (p) in the PACF, and the MA can be identified in significant autocorrelation at lag (q) in the ACF. The parameters P and Q correspond to the determination of p and q. Finally, seasonal patterns are recognized as periodic spikes which appear at multiples of the seasonal lag (S). These steps help systematically determine the appropriate parameters for constructing SARIMA models.

XGboost model

XGBoost is a boosting algorithm proposed by Chen Tianqi et al. in 2016 [26]. Based on the gradient boosting framework, it sequentially trains multiple weak classifiers, minimizes the loss function through gradient descent, and continuously enhances the model's predictive ability. Figure 2 illustrates the basic structure of the XGBoost model. Initially, input features are constructed using the hysteresis method, and a simple prediction model is initialized. Subsequently, the model calculates the error between its predictions and the training set, adjusts parameters based on this error, and constructs a new model. This iterative process constructs multiple models sequentially, gradually improving the performance of a single model. Finally, all models are integrated and weighted according to their contribution to the overall prediction effect, resulting in the final XGBoost model.

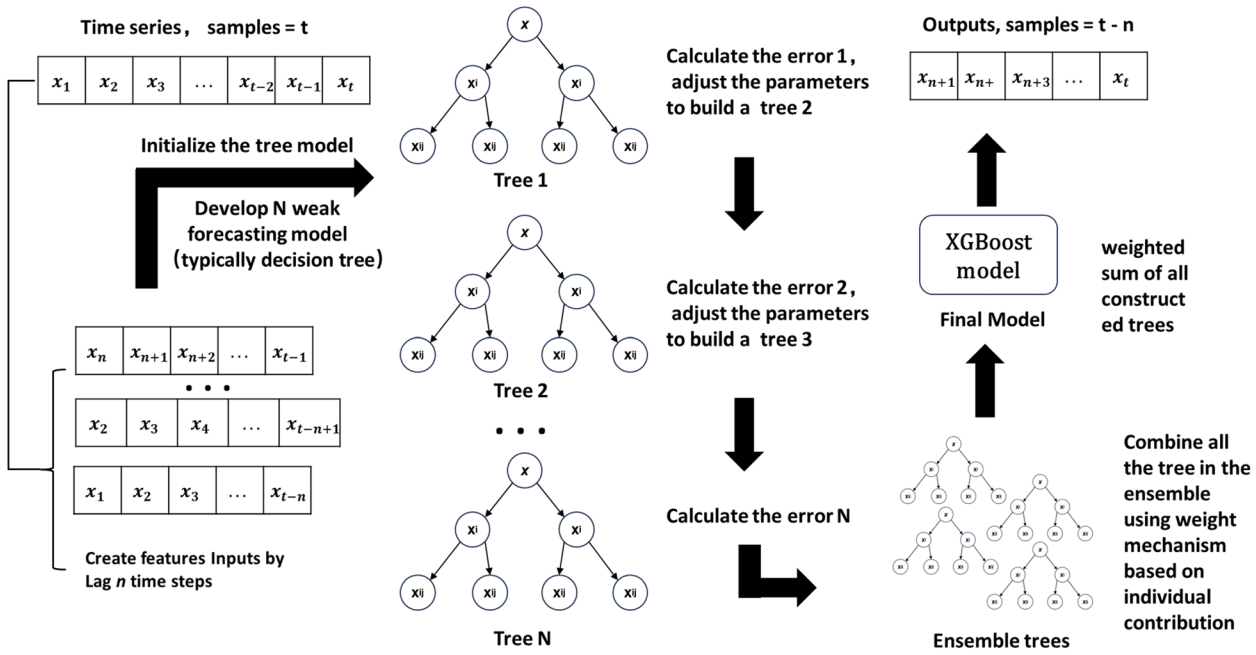

Fig. 2 Architecture of the extreme gradient boosting

Deep learning models

Data pre-processing

In neural network models, training process, large differences in data scales can lead to excessively large or small gradient updates. Additionally, outliers in the data may cause certain features to dominate others. Normalization ensures that all features are on a similar scale, reduces the impact of outliers, and enhances the model's convergence speed and generalization ability [27]. Here, original data were normalized with the formula provided below:

$$X = \frac{x - x_{min}}{x_{max} - x_{min}}$$

where the X is the normalized value of the data, the x is the original data, and x_{max} and x_{min} are their maximum and minimum values, respectively.

Furthermore, the input data for these models needed to contain both inputs (X) and outputs (Y), and needed to be reconstructed into 3D tensors format (samples, timesteps, features). Given that the ILI% data were a one-dimensional dataset of length n, a sliding-window approach was employed (Fig. 3). Observations of the current k time points consisted of the input data X, and observations of the current time points consisted of the output data Y. We then used the Numpy library in Python to convert data into an (n, n-k, 1) format.

CNN model

CNN models have superior capabilities in extracting local features [28]. They can be employed in computer vision and time series prediction and analysis [29]. CNN models consist of convolutional layers, pooling layers, activation functions, and dense layers (Fig. 4). For our time series data, 1D kernels were used to slide across the data (x) and compute dot products between the kernel and local data at each step, eventually generating feature maps. We then used max pooling on feature maps to reduce dimensions while retaining important features. After adding nonlinearity to pooled data using a tanh function, data were input into the dense layer and produced a CNN model using the following formula:

$$\begin{aligned} X &= (x * w) + b_i \\ X_c &= max\left(X_{i:i+k-1}\right) \\ y &= W * X_c + b_j \end{aligned}$$

where $(x * w)$ denotes the convolution operation, b is a bias term $X_{i:i+k-1}$ is a set of consecutive elements in the input feature sequence, $max()$ is a maximum extraction operation, W is a weight matrix, and $*$ is a matrix multiplication operation.

LSTM model

LSTM models are improved neural networks based on RNN [30]. They have a similar structure to RNN, but

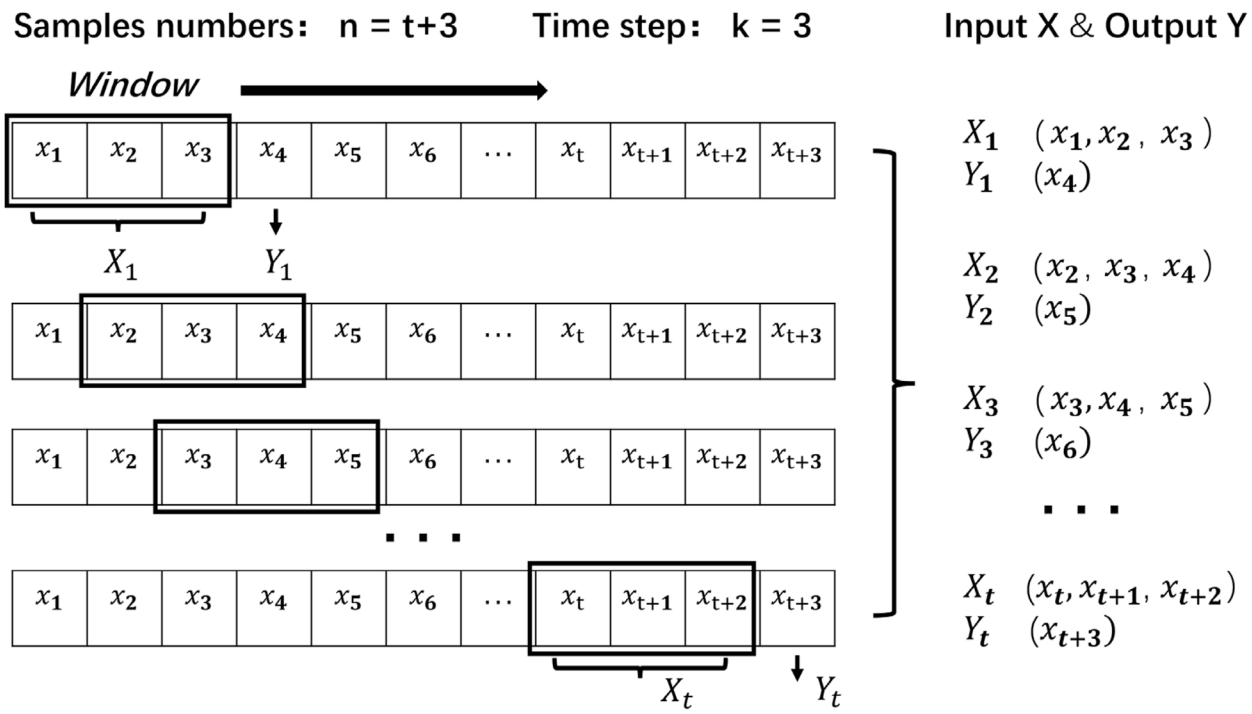

Fig. 3 Sliding window method to generate inputs (X) and outputs (Y)

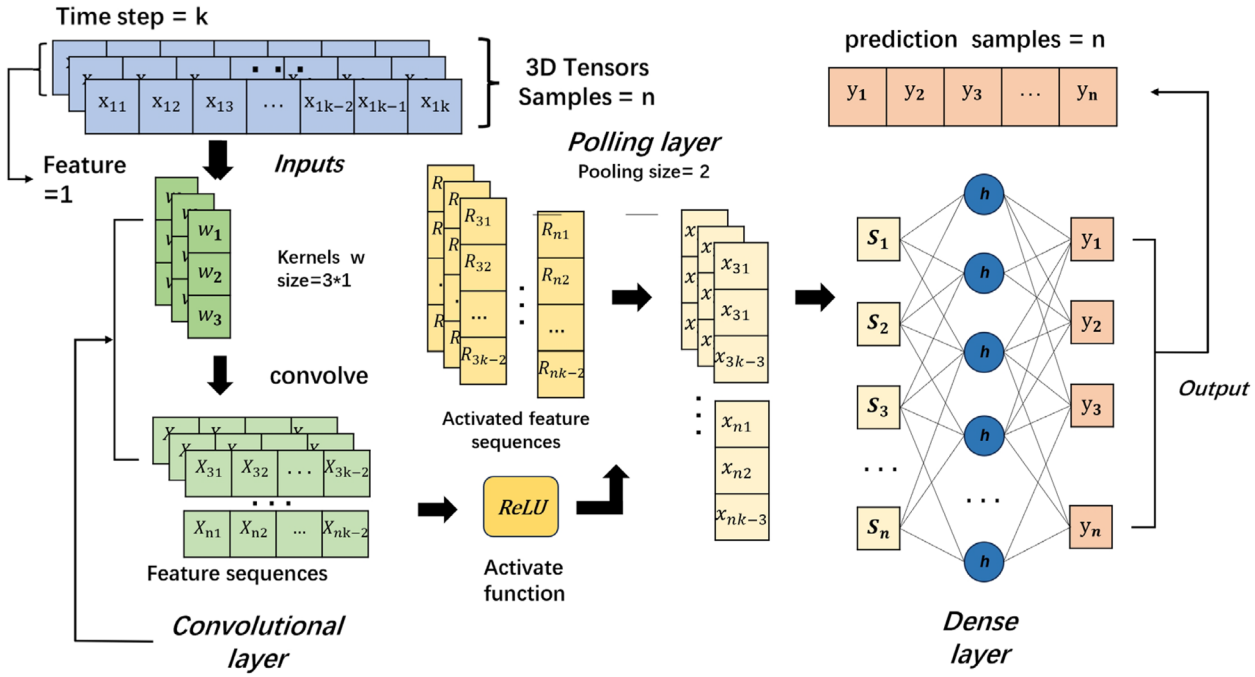

Fig. 4 Architecture of the convolutional neural network

enhanced memory units (Figs. 5 and 6) [31]. By introducing gate control mechanisms, LSTM models address the long-period dependency problems and the gradient vanishing and exploding issues that are present in RNN models. Our LSTM memory units consisted of input (I), forget (F), and output (O) gates, where the F gate decided the discording

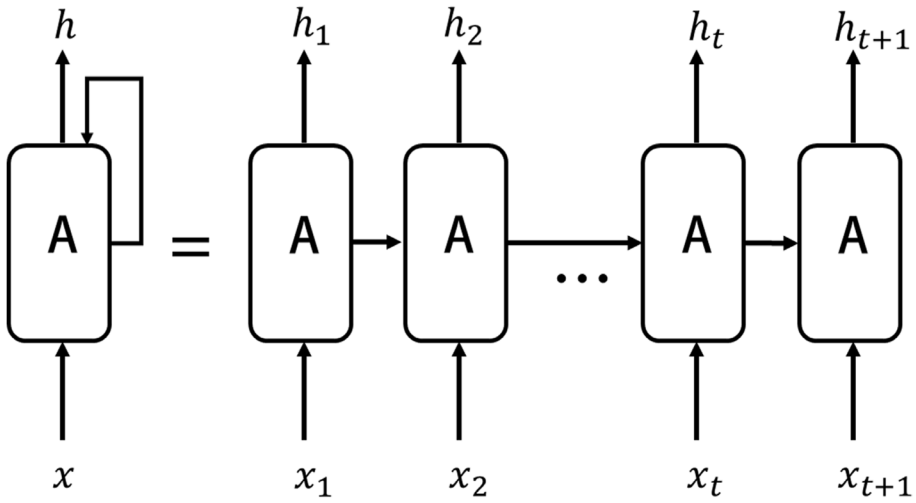

Fig. 5 Neural units of the recursive neural network

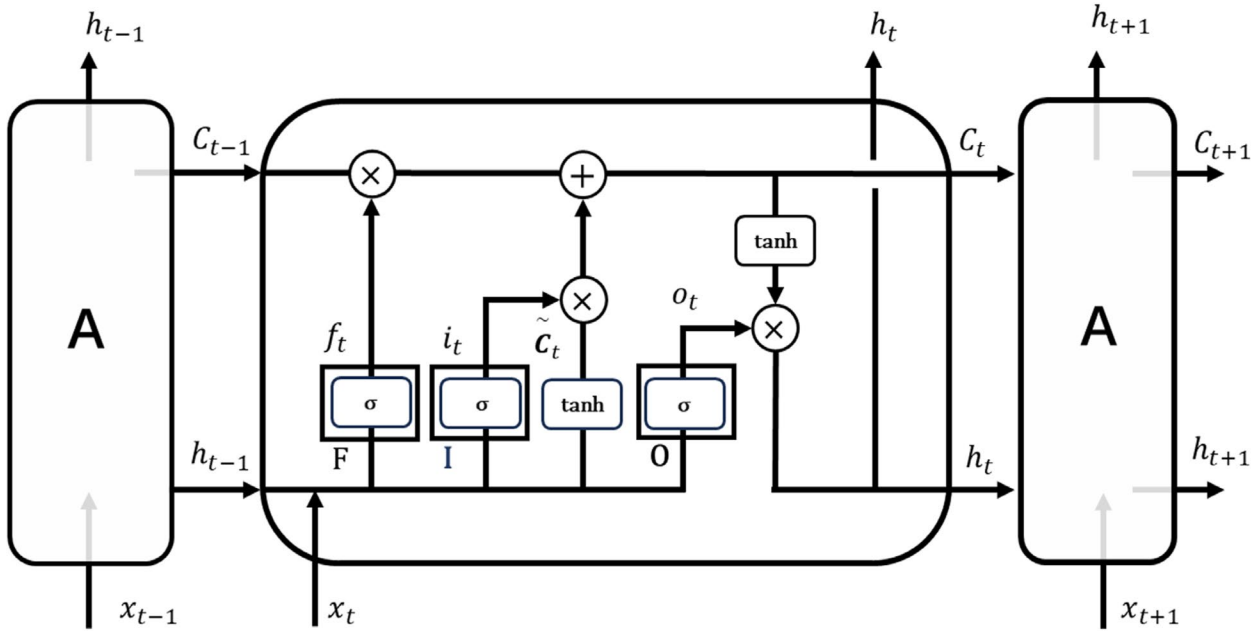

Fig. 6 Memory cells of the long-short term memory neural network

and preserving of information, the I gate decided what new information to store in the cell state, and the O gate decided what the next hidden state should be. The calculated LSTM formula was as follows [32]:

$$\begin{cases} cf_t &= \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \widetilde{C}_t &= tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * \widetilde{C}_t \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t * tanh(C_t) \end{cases}$$

where W_C is the weight matrix, and b is the bias term.

The F gate takes the previous hidden state h_{t-1} and current input $\boldsymbol{x_t}$ as input, passes a sigmoid activation function, and outputs f_t. In the I gate, the σ layer outputs i_t and the tanh layer creates a vector of new candidate values$\widetilde{\boldsymbol{C}}_t$, and the cell state C_t vector is updated by combiningf_t,i_t, and$\widetilde{\boldsymbol{C}}_t$. Finally, in the O gate, It filters the input information by passing it through the σ layer with the output o_t,ando_t, along with the C_t that has been operated on by tanh layer, completes the update ofh_{t+1}.

CNN-LSTM model

Our novel hybrid CNN-LSTM model combined the local feature extraction capability of CNN models and the long-term dependency resolution of LSTM models [33, 34]. Figure 7 visualizes the architecture of our CNN-LSTM model, which was composed of convolutional layers, pooling layers, LSTM layers, the tanh activation function, and a dense layer. The convolutional layers were encoders that used 1D kernels to extract features from the original data and convert them into feature maps. The pooling layer selected and reduced the dimensions of the feature maps, which were then decoded by the LSTM layers and fed into the dense layer to get the model output. This process generated the following formula:

Model construction and validation

The SARIMA model was constructed using the "forecast" package based on R 4.2.3. The XGBoost model was constructed using the "xgboost" and "sklearn" libraries in Python 3.9. Additionally, the CNN, LSTM, and CNN-LSTM models were developed using the PyTorch 2.2.1 deep learning framework based on Python 3.9.

After the models were constructed, they need to be validated and their parameters determined. For the SARIMA model, the prediction performance varies based on different parameter combinations. By analyzing the Autocorrelation Function (ACF) plot and the Partial Autocorrelation Function (PACF) plot, appropriate values of p, q, P, and Q can be identified. Typically, these values do not exceed 2. Therefore, we can construct multiple alternative models using an exhaustive method to find the best fit. Subsequently, the residuals of the model undergo a Box-Ljung test to identify outliers and assess the distribution. Finally, the model with the lowest Akaike Information Criterion (AIC) value is selected as the optimal model.

For models such as XGBoost, CNN, LSTM, and hybrid CNN-LSTM models, their performance is also influenced by hyperparameters. During the model training phase, the first step involves determining the range of hyperparameters using the training set. Next, the training set is randomly divided into k subsets using the K-fold cross-validation method, where k-1 subsets are used for training and the remaining subset is used for validation. The grid search method iterates through various hyperparameter combinations, training a model on each subset, and calculating the average loss across all k subsets. Finally, the hyperparameter set that yields the smallest average loss is chosen to build the best-performing model.

Comparison of model predictive effects

After the models are constructed and validated, the testing set is used to make retrospective predictions. The mean absolute error (MAE), root mean squared error (RMSE), mean absolute percentage error (MAPE), coefficient of determination (R^2), correlation coefficient (r),

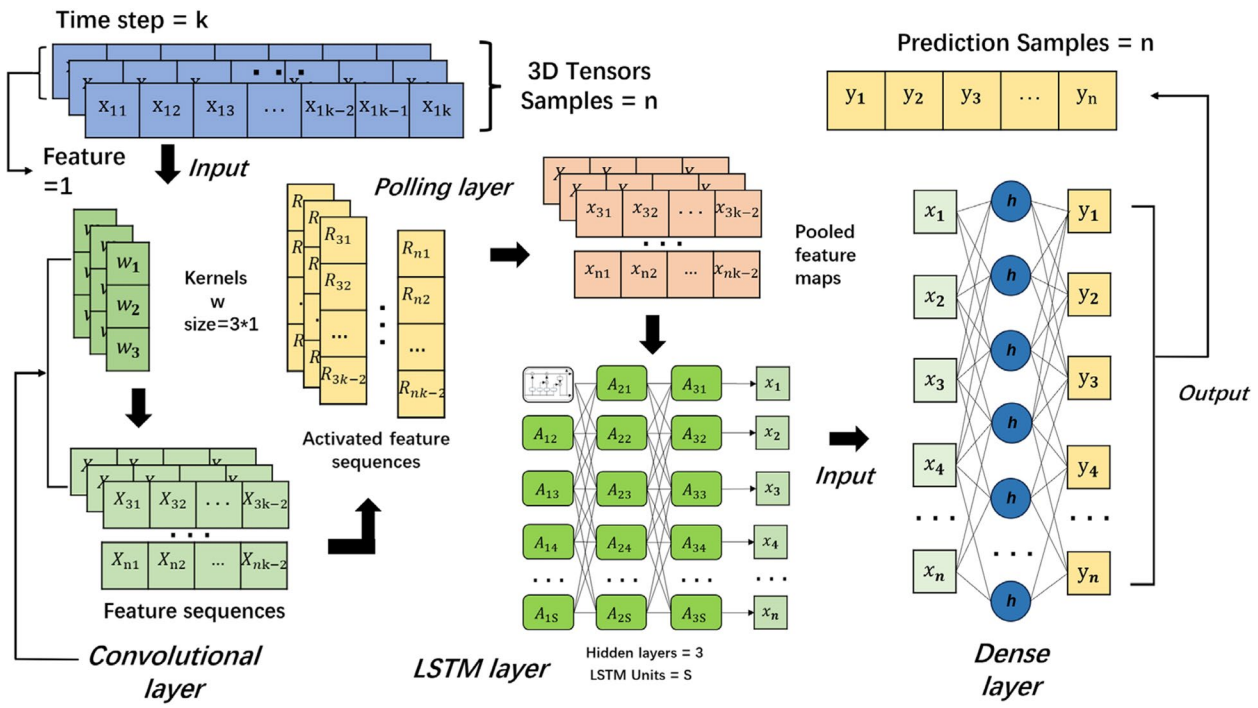

Fig. 7 Hybrid model framework

and explained variance (*Evar*) were calculated to evaluate the predictive performance of each model against the true values. The formulas used for calculation were as follows:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|x_i - \widehat{x}_i\right|$$
$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_i - \widehat{x}_i\right)^2}$$
$$MAPE = \frac{100\%}{N}\sum_{i=1}^{N}\left|\frac{x_i - \widehat{x}_i}{x_i}\right|$$
$$R^2 = 1 - \frac{\sum_{i=1}^{N}(x_i - \widehat{x}_i)^2}{\sum_{i=1}^{N}(x_i - \overline{x})^2}$$
$$r = \frac{\sum_{i=1}^{N}((x_i - \overline{x}) - (y_i - \overline{y}))^2}{\sqrt{\sum_{i=1}^{N}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{N}(y_i - \overline{y})^2}}$$
$$Evar = 1 - \frac{\sum_{i=1}^{N}\left((x_i - \widehat{x}_i) - E(\overline{x} - \widehat{x}_i)\right)^2}{\sum_{i=1}^{N}(x_i - \overline{x})^2}$$

where x_i represents the true value at time i, $\widehat{x}_i$ is the predicted value of the model at time i, N is the sample size at the time of testing, and $\overline{x}$ is the mean value of the sequence x.

Results

Temporal distribution of ILI%

The fluctuation in ILI and ILI% in Hebei Province between 2010 and 2022 are demonstrated in Figs. 8 and 9. The changes in both are essentially identical. From 2010 to 2019, the seasonal distribution of ILI% showed a unimodal pattern, with the peak of the epidemic in the winter and spring of each year, starting from the 40th week of the first year and continuing to the 15th week of the following year. Between 2020 and 2022, in addition to the peak of the epidemic in winter and spring, there was a small ILI% peak in the spring and summer. Between 2017 and 2019, the ILI% prevalence peak increased year over year, returned to fluctuating levels between 2020 and 2021, and reached a new peak at the end of 2022.

Data partitioning

Our 2010 to 2022 data spanned 676 weeks. We thus segmented it in the 21st week of 2020 to represent an 80–20 split. The initial 80% of the data, or 540 weeks, was used as the training set, and the remaining 20%, or 136 weeks, was used as the test set. Results are shown in Fig. 10.

SARIMA model

Previous research suggests that ILI has a noticeable seasonality Fig. 7. Specifically, ILI exhibits a seasonal cycle of 52. ADF testing of the training dataset revealed a lack of sequence stationarity. However, after applying first-order seasonal differencing, the sequence became stationary and passed both the ADF and white noise tests. Thus, data were successfully transformed into a stationary white noise sequence suitable for the development of the SARIMA model (Table 1).

After determining the values of d and D through the previously-mentioned differencing operation, we initially determined the values of p, P, q, and Q by examining the

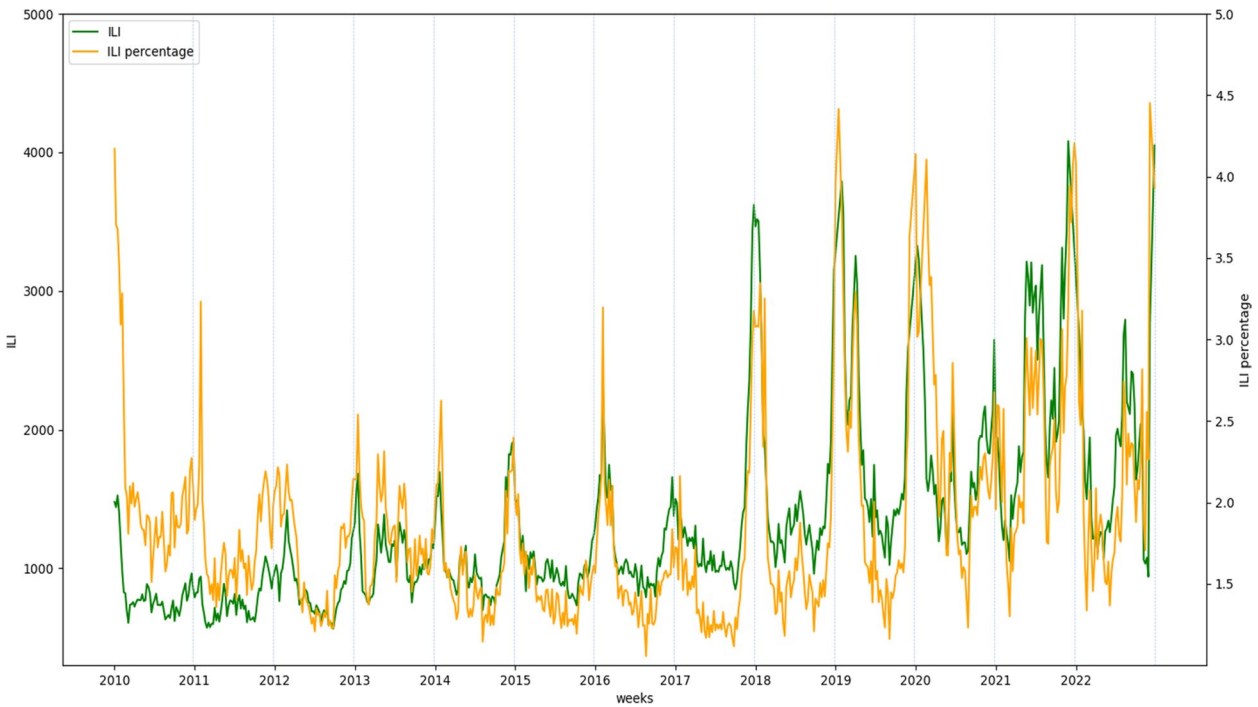

Fig. 8 ILI and ILI% activity from 2010 to 2022

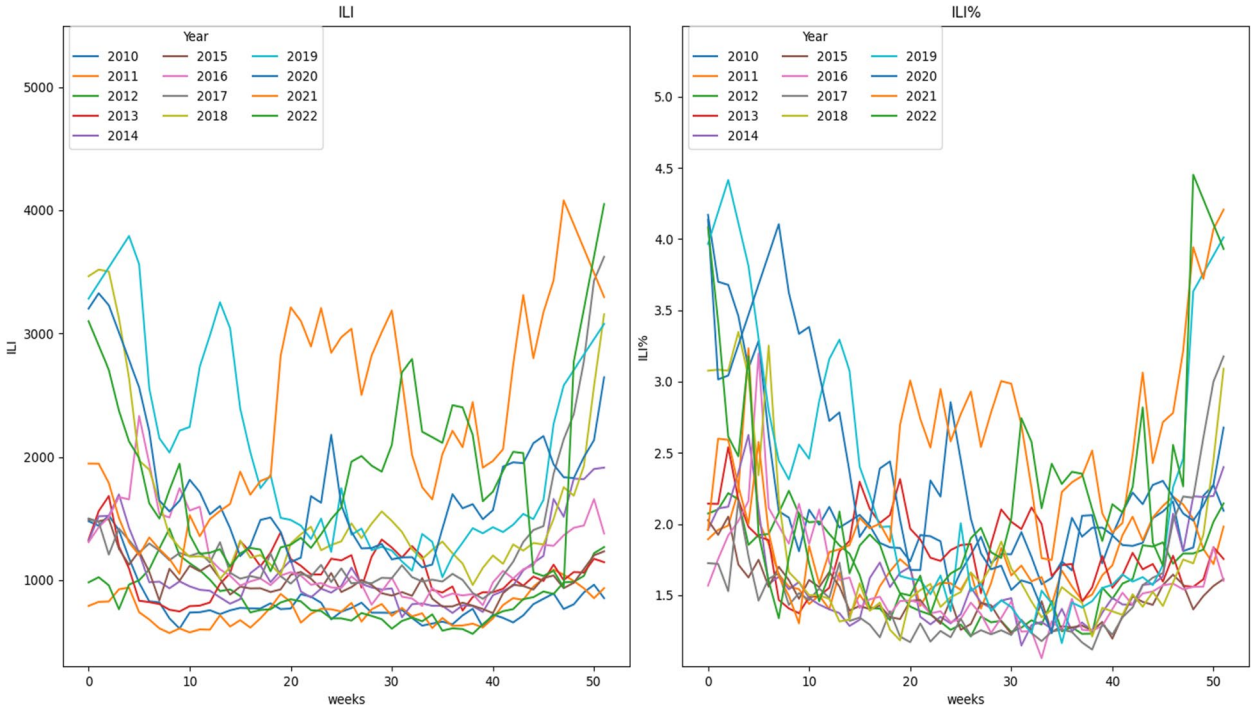

Fig. 9 Annual changes in ILI and ILI% from 2010 to 2022

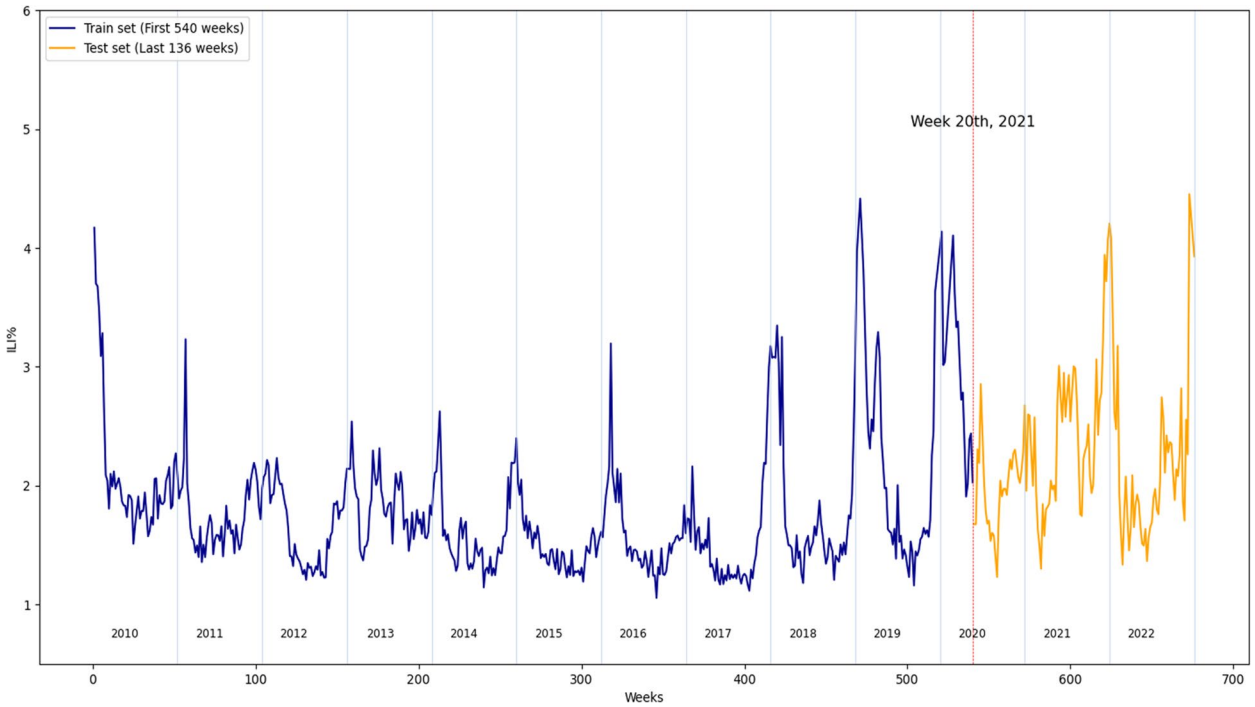

Fig. 10 Data division into training and testing sets

ACF and PACF plots of the differenced series (Fig. 11). Various alternative models were constructed by combining different parameters (Table 2). The best model of those that passed the parameter tests, selected following the principle of minimizing AIC, was the SARIMA (2, 0, 2) $(1, 1, 1)_{52}$ model. Residual testing showed that residuals

Table 1 Sequence stationary and white noise test results

Sequences	ADF		Box-Ljung	
	t	*P*	$\chi 2$	*P*
Raw sequence	-1.92	0.0547	1569.0	< 0.01
Differenced sequence	-215.3	< 0.01	1723.6	< 0.01

at different lag orders exhibited white noise sequences, indicating effective use of sequence information (Fig. 12).

XGBoost model

The construction process of an XGBoost model can be divided into three parts: feature engineering, model construction, and model validation. Since the ILI% data is one-dimensional, we use the current value as the output and construct input features by lagging one seasonal period (52 weeks) before building the XGBoost model. During the model building phase, we utilize the XGBRegressor class to establish a gradient boosted tree regression model with mean squared error as the loss function.

The alternative hyperparameters for the XGBoost model include the maximum depth of the tree (3, 5, or 7), the learning rate (0.01, 0.1, or 0.2), the number of decision trees (100, 200, or 300), and the proportion of subsamples (0.7, 0.8, or 0.9). Through fivefold cross-validation using the grid search method provided by the sklearn library, we determined the optimal combination of hyperparameters: maximum tree depth of 7, learning rate of 0.2, 200 trees, and subsample proportion of 0.7, yielding a minimum average error of 0.1173. The results are shown in Fig. 13. Finally, we built an XGBoost model based on the best hyperparameters described above.

CNN, LSTM, and CNN-LSTM models

The ILI% period of 52 led us to select a time step of 52 for data reconstruction using the sliding window method. Following reconstruction, the training set had a data shape of (488, 52, 1) and the test set had a data shape of (84, 1).

The development process of neural network models involved into three parts: model definition, validation, and training. During the models definition phase, we

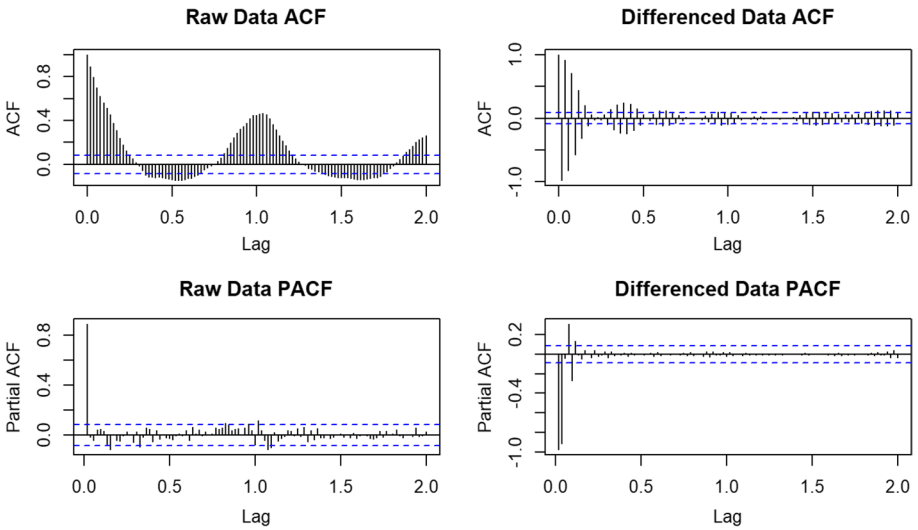

Fig. 11 Sequence' autocorrelation function and partial autocorrelation function

Table 2 Alternative model parameters and test results

Models	Model Parameters						AIC	Box-Ljung	
	AR_1	AR_2	MA_1	MA_2	SAR_1	SMA_1		x^2	*P*
$(1,0,0)(1,1,0)_{52}$	0.7764	—	—	—	-0.5864	—	23.19	9.0507	0.0598
$(1,0,1)(1,1,0)_{52}$	0.8765	—	-0.1871	—	-0.5727	—	20.46	2.7802	0.5952
$(1,0,0)(1,1,1)_{52}$	0.8342	—	—	—	-0.2246	-0.6990	-10.41	14.7026	0.0536
$(2,0,2)(1,1,0)_{52}$	0.0879	0.7016	0.6089	-0.1872	-0.5728	—	-5.75	1.8434	0.7645
$(2,0,2)(1,1,1)_{52}$	-0.0431	0.9702	0.7337	-0.2663	-0.1743	-0.8272	-17.02	2.3738	0.6674

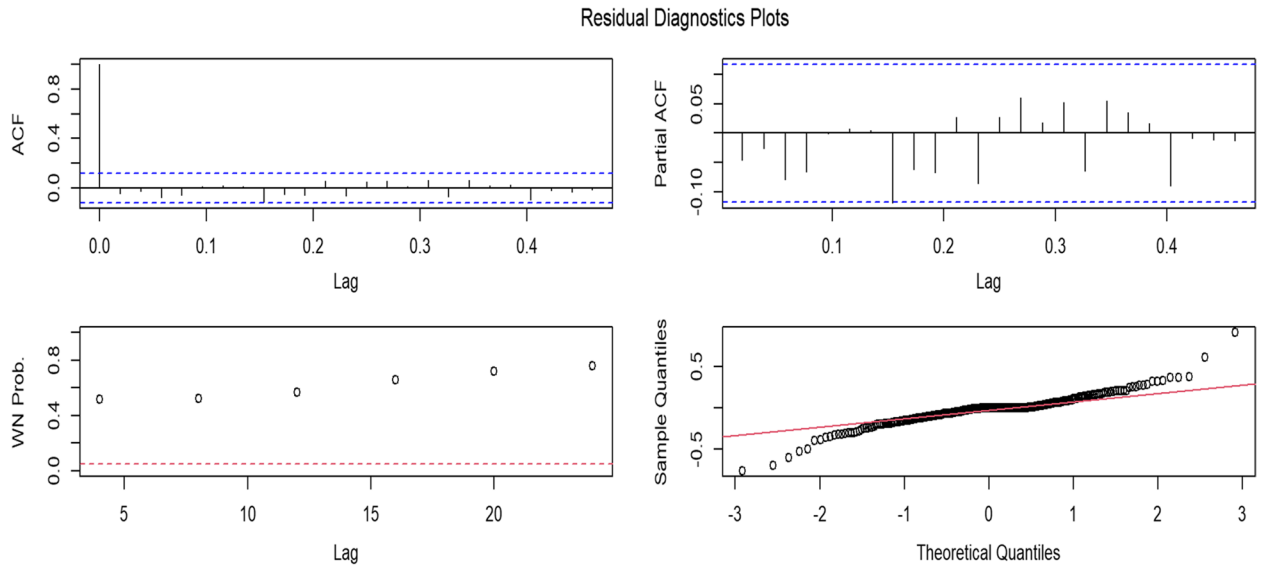

Fig. 12 SARIMA model residual test results

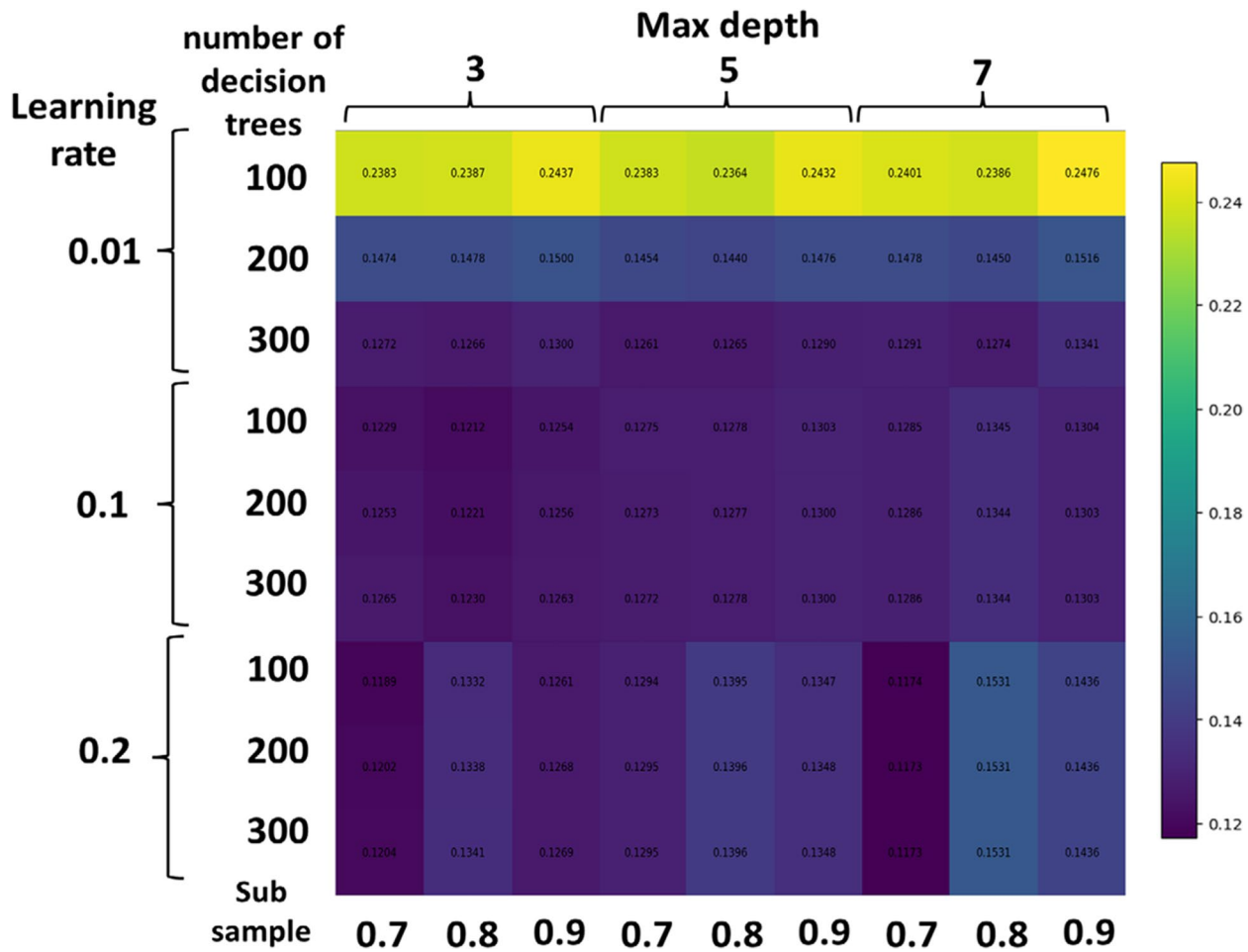

Fig. 13 XGBoost's losses for different combinations of hyperparameters

defined the model structures and forward propagation modules based on the PyTorch platform. Specifically, the CNN model includes convolutional layer and pooling layer, dropout layer, and fully connected layer with the ReLU activation function. The LSTM model consists of LSTM layer, dropout layer, and fully connected layer. The CNN-LSTM model combined aspects from both the CNN and LSTM models by incorporating convolutional layer, LSTM layer, dropout layer, and fully connected layer. In the forward propagation modules: For the CNN model, the tensor input passes through the convolutional and pooling layers, followed by activation functions and dropout layers, culminating in the fully connected layer to produce the output. The LSTM model generates output through its LSTM layer, dropout layer, and fully connected layer. The CNN-LSTM model processes the feature map through a convolutional layer, LSTM layer, dropout layer, and fully connected layer to obtain the final output.

In the model validation phase, alternative hyperparameters for the CNN model include the number of convolutional layers (1 or 2), convolution kernel size (3 or 5), number of pooling layers (1 or 2), learning rate (0.01 or 0.001), and batch size (16, 32, or 64). For the LSTM model and CNN-LSTM model, hyperparameters include the number of LSTM layers (1, 2, or 3), number of hidden units (16, 32, or 64), learning rate (0.01 or 0.001), and batch size (32, 64, or 128). After fivefold cross-validation using the grid search method, optimal hyperparameter combinations were determined: The CNN model consisted of 2 convolutional layers with a kernel size of 5 and 1 pooling layer, using a learning rate of 0.001 and a batch size of 16, achieving a minimum average loss of 0.0038. The LSTM model included 2 LSTM layers with 64 hidden units, a learning rate of 0.01, and a batch size of 128, achieving a minimum average loss of 0.0031. Additionally, the CNN-LSTM model featured 1 LSTM layer with 16 hidden units, a learning rate of 0.001, and a batch size of 32, achieving a minimum average loss of 0.0021. These results are illustrated in Fig. 14. Finally, three models were constructed using the best combination of hyperparameters.

In the model training phase, which includes backward propagation and iterative training. MSE served as the loss assessment metric and Adam as the optimizer. After 200 training epochs, the losses converged: CNN model loss was 0.0005, LSTM model loss was 0.0010, and CNN-LSTM model loss was 0.0014. The training loss of three models are shown in Fig. 15.

Comparison and analysis of models

After constructing the models, we utilized the aforementioned five models to make retrospective predictions (Fig. 16) and compared them with the test set to calculate MAE, RMSE, MAPE, R^2, r, and *Evar*. The SARIMA model was used to forecast ILI% over a span of 136 weeks starting from week 21 in 2020. The XGBoost model predicts 84 weeks based on the constructed input features (starting from week 21 in 2021). And the neural network models were used to generate forecasts for the subsequent 84 weeks (starting from week 21 in 2021) via test inputs created by the sliding-window method. All the models demonstrated improved prediction accuracy with regard to ILI% tendencies. We evaluated the models' predictive performance using MAE, RMSE, MAPE, R^2, r, and *Evar*, with specific results detailed in Tables 3, 4, 5 and 6. On the testing set, the SARIMA model had the highest MAE, RMSE, and MAPE, indicating relatively poor prediction performance in terms of R^2, r, and *Evar*. Conversely, the CNN, LSTM, XGBoost, and CNN-LSTM models showed smaller prediction errors and higher R^2, r, and *Evar*. Among these models, prediction errors ranked from largest to smallest as CNN, LSTM, XGBoost, and CNN-LSTM, while R^2, r, and *Evar* followed as CNN, XGBoost, LSTM, and CNN-LSTM, respectively. Overall, CNNs demonstrated the weakest

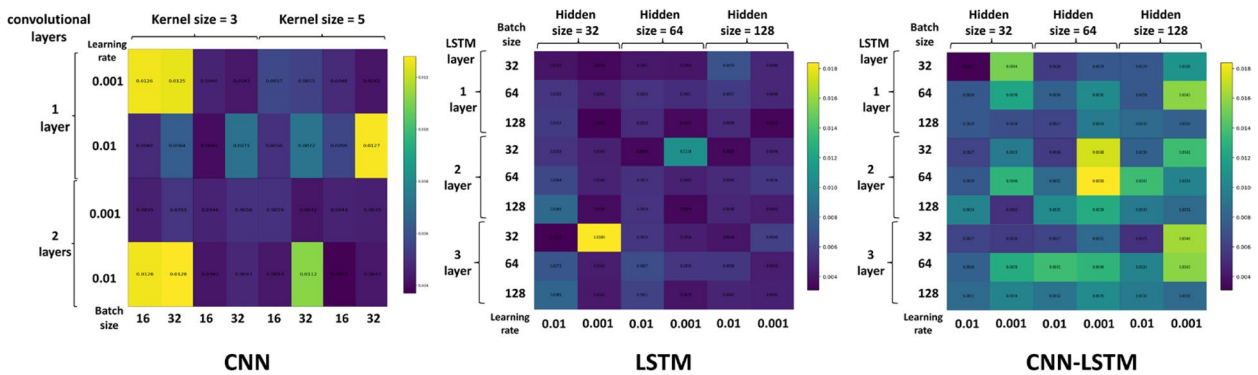

Fig. 14 Losses of distinct hyperparameter combinations in three models

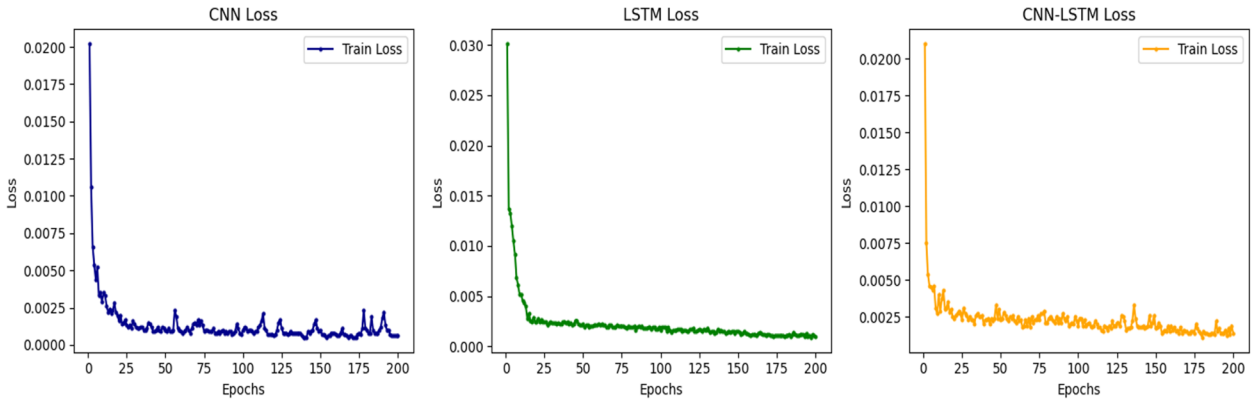

Fig. 15 Change in training loss over 200 epochs of the three models

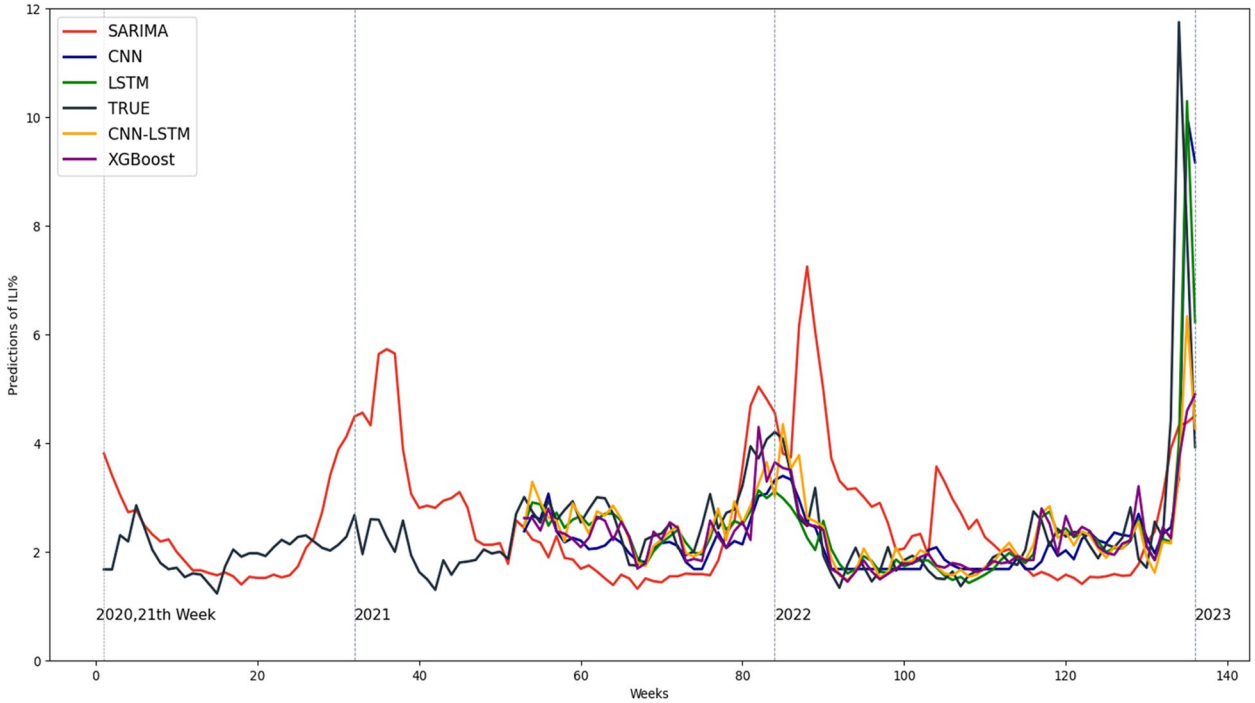

Fig. 16 Comparison of forecasting results from all four five models

Table 3 Mean Absolute Errors across different models

Models	MAE	MAE Changes Compared to				MAE Improvements Compared to			
		SARIMA	XGBoost	CNN	LSTM	SARIMA	XGBoost	CNN	LSTM
SARIMA	0.8917	—	0.4088	0.3243	0.3758	—	-84.66%	-57.16%	-72.84%
XGBoost	0.4829	-0.4088	—	-0.0845	-0.0330	45.85%	—	14.89%	6.40%
CNN	0.5674	-0.3243	0.0845	—	0.0515	36.37%	-17.50%	—	-9.98%
LSTM	0.5159	-0.3758	0.0330	-0.0515	—	42.14%	-6.83%	9.08%	—
CNN-LSTM	0.4529	-0.4388	-0.0300	-0.1145	-0.0630	49.21%	6.21%	20.18%	12.21%

Table 4 Root Mean Squared Errors across different models

Models	MAE	MAE Changes Compared to				MAE Improvements Compared to			
		SARIMA	XGBoost	CNN	LSTM	SARIMA	XGBoost	CNN	LSTM
SARIMA	1.3933	0.0000	0.3229	0.1698	0.2463	0.00%	-30.17%	-13.88%	-21.47%
XGBoost	1.0704	-0.3229	0.0000	-0.1531	-0.0767	23.18%	0.00%	12.52%	6.68%
CNN	1.2235	-0.1698	0.1531	0.0000	0.0765	12.19%	-14.31%	0.00%	-6.67%
LSTM	1.1470	-0.2463	0.0767	-0.0765	0.0000	17.67%	-7.16%	6.25%	0.00%
CNN-LSTM	1.0385	-0.3548	-0.0319	-0.1851	-0.1086	25.47%	2.98%	15.13%	9.47%

Table 5 Mean Squared Percentage Errors across different models

Models	MAPE	MAE Changes Compared to				MAE Improvements Compared to			
		SARIMA	XGBoost	CNN	LSTM	SARIMA	XGBoost	CNN	LSTM
SARIMA	41.79%	—	26.6914	24.1035	25.2066	—	-177%	-136%	-152%
XGBoost	15.10%	-26.6914	—	-2.5879	-1.4849	64%	—	15%	9%
CNN	17.69%	-24.1035	2.5879	—	1.1030	58%	-17%	—	-7%
LSTM	16.58%	-25.2066	1.4849	-1.1030	—	60%	-10%	6%	—
CNN-LSTM	14.58%	-27.2063	-0.5149	-3.1028	-1.9998	65%	3%	18%	12%

Table 6 Other metrics across different models

Metrics	Models				
	SARIMA	XGBoost	CNN	LSTM	CNN-LSTM
R2	0.1778	0.3811	0.1913	0.4078	0.4174
Evar	0.1177	0.4177	0.2227	0.4260	0.4359
r	0.4621	0.6213	0.5734	0.6649	0.6720

forecasting performance, whereas CNN-LSTM showed the most accurate predictions. Across the entire testing set, compared to the SARIMA model, the CNN-LSTM model exhibited reductions in MAE, RMSE, and MAPE by 0.4388, 0.3548, and 27.2063%, respectively, resulting in corresponding accuracy improvements of 49.21%, 25.47%, and 65%. When compared with the XGBoost model, these reductions were 0.0300, 0.0319, and 0.5149%, respectively, with corresponding improvements

Table 7 Evaluation metrics of five models in different prediction scales

Models	Scale (weeks)	MAE	RMSE	MAPE(%)	R^2	*r*	*Evar*
SARIMA	4	0.6233	0.6739	21.8783	-0.6032	0.0845	-0.9007
	26	0.7755	0.8569	30.0461	-0.3985	0.4624	-0.4158
	52	1.0275	1.3660	45.8447	0.2555	0.3180	0.2549
XGBoost	4	0.2006	0.2278	7.0144	0.5043	0.8149	0.6621
	26	0.3248	0.3810	12.8012	0.0116	0.6175	0.3583
	52	0.3446	0.4485	13.2950	0.6060	0.8265	0.6830
CNN	4	0.2209	0.3236	7.5140	-1.0358	0.0953	-1.3523
	26	0.3937	0.4973	14.8261	0.1684	0.4979	0.1673
	52	0.4042	0.5233	15.1939	0.4636	0.8115	0.6387
LSTM	4	0.3732	0.3966	13.1441	-0.5581	-0.4902	-0.3480
	26	0.3104	0.3551	12.3784	0.1416	0.5389	0.2230
	52	0.3848	0.4884	14.8699	0.5329	0.7944	0.5968
CNN-LSTM	4	0.4481	0.4595	15.9829	-0.5119	-0.4741	-0.5111
	26	0.3115	0.3688	12.2882	0.0740	0.5535	0.1306
	52	0.3600	0.4470	14.4859	0.6086	0.7891	0.6177

of 6.21%, 2.98%, and 3%, respectively. The specific results are shown in Table 3.

We used five models for predictions over the next 4 weeks, 26 weeks, and 52 weeks, and calculated prediction errors, R^2, *r*, and *Evar*. These specific results of these are shown in Table 7. As prediction length increased, so did prediction errors for all models (Fig. 17). At each prediction interval, the SARIMA model consistently showed the largest prediction error, accompanied by lower R^2, *r*, and *Evar*. In predictions for the next 4 weeks, the XGBoost model exhibited the smallest prediction error, followed by CNN and LSTM, with CNN-LSTM showing the largest error. Over 26 weeks, CNN-LSTM had the smallest error, followed by XGBoost and LSTM, while CNN had the largest error. Looking ahead to 52 weeks, CNN-LSTM maintained the smallest error, followed by XGBoost and LSTM, with CNN demonstrating the largest error. It's important to note that R^2, *r*, and *Evar* are influenced by sample size, with these metrics generally showing an initial increase followed by a decrease across different prediction periods for the five models. For instance, in predictions over the next 4 weeks, the XGBoost model showed the highest r and Evar values, followed by LSTM, CNN-LSTM, and CNN, while LSTM exhibited the highest R-squared, followed by XGBoost and CNN-LSTM, and CNN showed the lowest values. Over 26 weeks, CNN-LSTM had the highest scores across all three metrics, followed by XGBoost, with CNN and LSTM close behind. Over 52 weeks, LSTM had the highest *r*, with CNN-LSTM and XGBoost close, while CNN-LSTM showed the highest R^2 and *Evar* values, followed by XGBoost and LSTM, with CNN recording the lowest scores.

Discussion

ILI surveillance is crucial for flu prevention and control efforts. By tracking ILI cases and using historical data to construct predictive models, deeper understandings of ILI fluctuation patterns can be obtained. These understandings allow for the implementation of proactive measures to reduce the impact of flu. In this study, we utilized historical ILI% data spanning from 2010 to 2023 to develop a CNN-LSTM hybrid model for retrospective forecasting. We examined and compared its predictive with the SARIMA model, XGBoost model, and standalone neural network models. The study revealed that all five models effectively captured the trend in ILI% changes and demonstrated strong predictive capabilities. Upon further analysis across varying forecast horizons, the CNN-LSTM model exhibited the highest forecasting accuracy, followed by the XGBoost model, LSTM model, and CNN model, with the SARIMA model performing the least accurately.

The SARIMA model is a widely used time series forecasting model that extrapolated future outcomes by integrating autocorrelation and moving average error results. Unlike the XGBoost and neural network models, SARIMA models do not rely on input features and

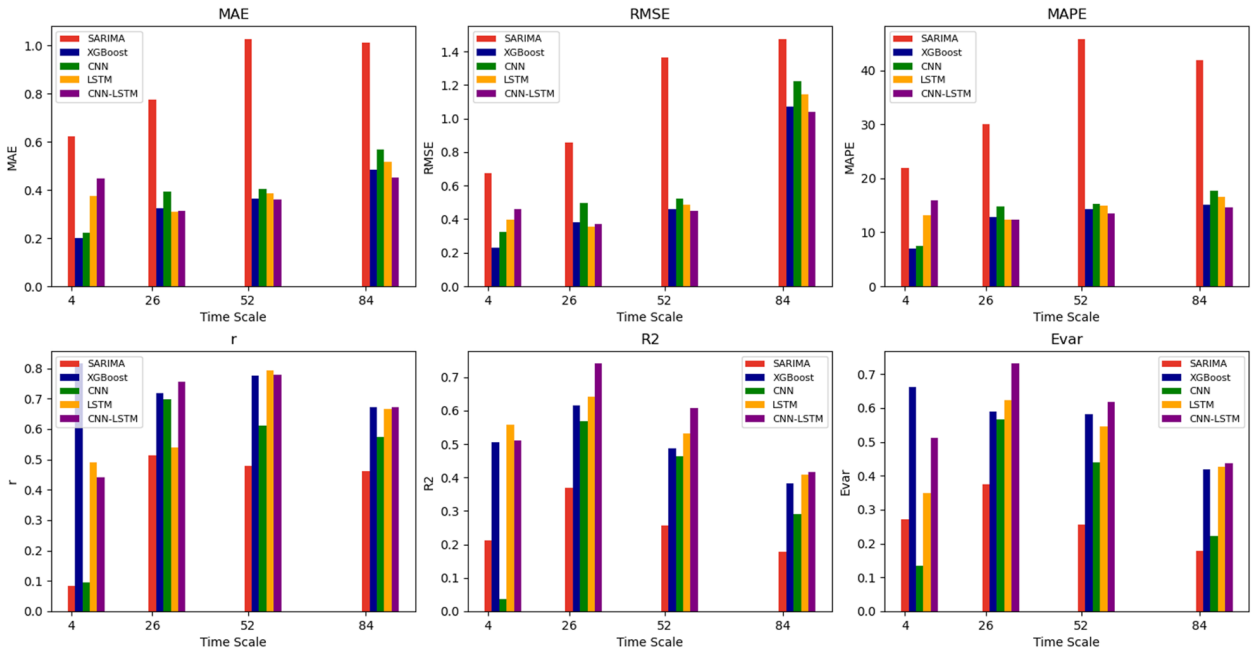

Fig. 17 Comparison of the evaluation metrics of the five models at different prediction scales

may offer superior forecasting ability for stationary time series. However, SARIMA models' predictive capabilities diminish when forecasting over long-term horizons. In Qiang Mao's study, SARIMA effectively predicted tuberculosis incidence in China for short-term forecasts but showed declining accuracy with longer prediction horizons [11]. Which have demonstrated SARIMA's superior accuracy in short-term prediction tasks. In our comparative study, SARIMA demonstrated poorer predictive capability too. In contrast, neural network models have no specific data requirements and have robust nonlinear mapping capabilities, enabling them to leverage more information for prediction [17]. This disparity underscores why the neural network models in our study had superior prediction accuracy compared to our SARIMA model. Research by S. Siami-Namini et al. indicates that SARIMA's structure, built on data autocorrelation and dependence, suits linear data prediction better than the other models examined [35]. SARIMA's performance on nonlinear data is inferior to that of machine learning and deep learning models, which aligns with our findings.

The XGBoost model is a highly popular machine learning technique renowned for its exceptional accuracy and robust generalization in sequence prediction. Utilizing an ensemble learning algorithm, XGBoost integrates multiple decision tree models through gradient boosting, continuously optimizing split points and leaf node divisions to minimize the loss function and enhance model performance. In our study, we employed the XGBoost model to forecast the test set across various time intervals, demonstrating strong predictive capabilities. Among the five models evaluated, XGBoost outperformed the LSTM, CNN, and SARIMA models, though it slightly lagged behind the CNN-LSTM hybrid model in accuracy. In research by Zheng-gang Fang et al., the XGBoost model effectively predicted Covid-19 case numbers in the United States and surpassed the ARIMA model in predictive accuracy, corroborating our findings [36]. Conversely, Junling Luo et al. found that the LSTM model exhibited superior accuracy over XGBoost in predicting Covid-19 cases in Canada [37]. In our study, while the LSTM model's prediction accuracy was slightly lower than XGBoost's, this difference may stem from our use of historical ILI% data exclusively for prediction. Notably, the CNN-LSTM model's predictive efficacy surpassed that of XGBoost in our study, suggesting that combining CNN and LSTM techniques holds promise for improving prediction performance.

CNN and LSTM are both deep learning models with distinct applications. CNN is commonly used in image and speech recognition, whereas LSTM excels in analyzing long time series data. Each model has unique advantages: CNN effectively learns sequence dynamics by extracting local patterns, while LSTM addresses long-term dependency issues using gate mechanisms within its memory units. The CNN-LSTM hybrid model combines these strengths and finds extensive use in fields such as finance and yield forecasting. In our study, CNN-LSTM was applied to forecast flu activity, outperforming individual models and ranking highest among the four models compared. In Muhammad, L. J et al.'s research, CNN-LSTM successfully predicted Covid-19 case numbers in South Africa and Botswana, achieving commendable accuracy [33]. Similarly, studies by Zhang, J et al. and Rui Yan et al. utilized CNN-LSTM to forecast air quality in Beijing, China, comparing it with CNN and LSTM models. Their findings consistently demonstrated CNN-LSTM's superior predictive performance [38, 39]. Notably, Rui Yan et al. also employed CNN-LSTM for multi-period prediction, aligning closely with our study's results.

limitations

Flu activity is influenced by various factors, including population immunity, conditions of transmission, and viral variability. In the retrospective projections of this study, the prediction accuracy of the five models declined to varying degrees during 2021 and 2022. This decline possibly due to the stringent preventative measures enacted during the COVID-19 pandemic. In Hu CY's study, many provinces and cities in China adopted preventive measures such as mask-wearing, closure of large public venues, and prohibition of large gatherings to control COVID-19. As COVID-19 spreads similarly to flu, activities related to flu and other respiratory infections were disrupted [40], and COVID-19 has similar clinical symptoms to influenza, and many COVID-19 patients go to the hospital to see an increase in the number of ILIs [41]. These will lead to significant bias in predictions. For future studies, data processing methods like singular spectrum analysis or one-hot encoding should be considered to mitigate the impact of COVID-19 prevention measures [42, 43].

In additional, since meteorological data such as meteorological data in the study area are not publicly available. Our study solely utilized ILI% data from 28 flu sentinel hospitals in Hebei Province. Although optimal hyperparameters were employed in model construction, the dimensions of the input features could potentially impact prediction results. In Athanasiou et al.'s study, multi-source data including monitoring, meteorological data, and Twitter search data were integrated, yielding

excellent prediction outcomes [44]. This suggests that incorporating more diverse and relevant data into models can enhance prediction accuracy. Future studies should consider leveraging multi-source data to improve overall model performance.

In infectious disease modeling, both linear and nonlinear relationships often coexist. Prediction methods for infectious diseases include linear models, machine learning models, and deep learning models, the latter of which possess strong nonlinear extraction capabilities. Although our study only utilized the combined advantages of CNN and LSTM to construct the CNN-LSTM model, there may be limitations in its ability to extract linear relationships. In Yiran Wan's prediction study on hand, foot, and mouth disease in Chongqing, China, the SARIMA-EEMD-LSTM model was employed, demonstrating higher prediction performance compared to single models [45]. This suggests that integrating different types of models could be beneficial for our future work.

Finally, it is crucial to provide a reasonable estimate of uncertainty in forecasting. As mentioned earlier, influenza incidence is influenced by many factors, and the predictive accuracy of the model can vary with changes in these uncertainties during actual predictions. It is essential to assess a reasonable confidence interval to model's predictive results. In a study by Morris, Michael et al., they utilized various neural network models to predict influenza activity and employed Gaussian distribution to estimate prediction result intervals, enhancing the realism and effectiveness of their predictions [46]. In our study, we did not adequately estimate the prediction uncertainty interval. Future research should address this by employing appropriate methods to estimate uncertainty factors, thereby incorporating confidence intervals into the model's prediction results to enhance effectiveness.

Conclusions

Seasonal flu epidemics impose a significant disease burden, emphasizing the need for scientifically accurate predictive models to facilitate early intervention. Our study developed a hybrid CNN-LSTM model to forecast the activation of flu in Hebei Province. We compared the predictive performance of the hybrid model against the SARIMA model, XGBoost model, CNN model, and LSTM model. Our analysis revealed that while all models effectively predicted ILI activity trends, the hybrid CNN-LSTM model demonstrated superior prediction accuracy compared to the others. These findings not only enhance the precision of flu prediction in Hebei Province but also demonstrate the efficacy of hybrid models in specific forecasting tasks. By providing timely scientific guidance before the flu season, proactive measures can be planned and implemented for prevention and control in Hebei Province. These findings not only enhance the precision of flu prediction in Hebei Province but also highlight the efficacy o f hybrid model in specific predictive tasks. By providing timely scientific guidance before the flu season, proactive measures can be planned and implemented for prevention and control in Hebei Province.

Abbreviations

ACF	Autocorrelation Function
ADF	Augmented Dickey-Fuller
AIC	Akaike Information Criterion
CNN-LSTM	Convolution Neural Network- Long Short Term Memory neural network
Evar	Explained Variance
ILI	Influenza-Like-Illness
LSTM	Long Short Term Memory neural network
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
PACF	Partial Autocorrelation Function
RMSE	Root Mean Square Error
RNN	Recursive Neural Network
r	Correlation Coefficient
SARIMA	Seasonal Auto-Regressive Indagate Moving Average
WHO	World Health Organization
σ	sigmoid activation function
XGBoost	Extreme Gradient Boosting
R^2	Coefficient of Determination

Acknowledgements

No applicable.

Authors' contributions

Guofan Li, Yan Li contributed equally to this work. Guofan Li was responsible for collecting the data, modeling, analyzing results, and writing of the initial draft; Yan Li was responsible for thesis guidance and review; Guangyue Han, Caixiao Jiang, Minghao Geng, Nana Guo, Wentao Wu was responsible for assistance of analysis and thesis guidance; Shangze Liu and Zhihuai Xing was responsible for assistance of data collection; Guangyue Han, Xu Han, Qi Li was responsible for work guidance, review, and funding support.

Funding

No applicable.

Availability of data and materials

These codes that support the findings of this study are openly available on GitHub at https://github.com/FWsfan/forecasting-models.git.
The data of ILI and outpatient/emergency department visits are protected and are not publicly available due to the data privacy laws of the Hebei Province Center for Disease Control and Prevention.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare no competing interests.

Received: 29 May 2024 Accepted: 25 July 2024
Published online: 12 August 2024

References

1. Ghebrehewet S, MacPherson P, Ho A. Influenza. BMJ. 2016;355:i6258. https://doi.org/10.1136/bmj.i6258.
2. Ryu S, Cowling BJ. Human Influenza Epidemiology. Cold Spring Harb Perspect Med. 2021;11(12). https://doi.org/10.1101/cshperspect.a038356.
3. Zhu AQ, Zheng YM, Qin Y, Liu SS, Cui JZ, Li ZL, et al. A systematic review of the economic burden of influenza in China. Zhonghua Yu Fang Yi Xue Za Zhi. 2019;53(10):1043–8.
4. Lei H, Yang L, Wang G, Zhang C, Xin Y, Sun Q, et al. Transmission patterns of seasonal influenza in China between 2010 and 2018. Viruses. 2022;14(9).https://doi.org/10.3390/v14092063.
5. Si X, Wang L, Mengersen K, Hu W. Epidemiological features of seasonal influenza transmission among 11 climate zones in Chinese Mainland. Infect Dis Poverty. 2024;13(1):4. https://doi.org/10.1186/s40249-024-01173-9.
6. Belongia EA, Kieke BA, Donahue JG, Greenlee RT, Balish A, Foust A, et al. Effectiveness of inactivated influenza vaccines varied substantially with antigenic match from the 2004–2005 season to the 2006–2007 season. J Infect Dis. 2009;199(2):159–67. https://doi.org/10.1086/595861.
7. Zimmerman RK, Nowalk MP, Chung J, Jackson ML, Jackson LA, Petrie JG, et al. 2014–2015 Influenza vaccine effectiveness in the United States by vaccine type. Clin Infect Dis. 2016;63(12):1564–73. https://doi.org/10.1093/cid/ciw635.
8. Yang X, Liu D, Wei K, Liu X, Meng L, Yu D, et al. Comparing the similarity and difference of three influenza surveillance systems in China. Sci Rep. 2018;8(1):2840. https://doi.org/10.1038/s41598-018-21059-9.
9. Lo C, Marculescu R. MPLasso: Inferring microbial association networks using prior microbial knowledge. PLOS Comput Biol. 2017;13(12):e1005915. https://doi.org/10.1371/journal.pcbi.1005915.
10. Tian C, Wang H, Luo X. Time-series modelling and forecasting of hand, foot and mouth disease cases in China from 2008 to 2018. Epidemiol Infect. 2019;147:e82. https://doi.org/10.1017/s095026881800362x.
11. Mao Q, Zhang K, Yan W, Cheng C. Forecasting the incidence of tuberculosis in China using the seasonal auto-regressive integrated moving average (SARIMA) model. J Infection Public Health. 2018;11(5):707–12. https://doi.org/10.1016/j.jiph.2018.04.009.
12. Fang Z-g, Yang S-q, Lv C-x, An S-y, Wu W. Application of a data-driven XGBoost model for the prediction of COVID-19 in the USA: a time-series study. BMJ Open. 2022;12(7):e056685. https://doi.org/10.1136/bmjopen-2021-056685.
13. Shajihan SAV, Wang S, Zhai G, Spencer BF Jr. CNN based data anomaly detection using multi-channel imagery for structural health monitoring. Smart Struct Syst. 2022;29(1):181–93. https://doi.org/10.12989/sss.2022.29.1.181.
14. Selvin S, Vinayakumar R, Gopalakrishnan E, Menon VK, Soman K, editors. Stock price prediction using LSTM, RNN and CNN-sliding window model. 2017 international conference on advances in computing, communications and informatics (icacci). Udupi: IEEE; 2017. https://doi.org/10.1109/ICACCI.2017.8126078.
15. Karevan Z, Suykens JA. Transductive LSTM for time-series prediction: an application to weather forecasting. Neural Netw. 2020;125:1–9. https://doi.org/10.1016/j.neunet.2019.12.030.
16. El Idriss T, Idri A, Abnane I, Bakkoury Z, editors. Predicting blood glucose using an LSTM neural network. 2019 Federated Conference on Computer Science and Information Systems (FedCSIS). Leipzig: IEEE; 2019.
17. Tsan YT, Chen DY, Liu PY, Kristiani E, Nguyen KLP, Yang CT. The prediction of influenza-like illness and respiratory disease using LSTM and ARIMA. Int J Environ Res Public Health. 2022;19(3). https://doi.org/10.3390/ijerph19031858.
18. Li Yan LY, Han GuangYue HG, Liu YanFang LY, Liu LanFen LL, Liu JingSheng LJ, Li Qi LQ, et al. Analysis of the results of influenza virus surveillance in Hebei Province from 2009 to 2015. 2015. https://doi.org/10.13350/j.cjpb.150815.
19. Zhang R, Jiang C, Li Y, Liu Y, Liu L, Han G, et al. Epidemiological characteristics and incidence trend of influenza like illness in Hebei, 2010–2020. Dis Surveill. 2022;37(11):1429–35. https://doi.org/10.3784/jbjc.20205070198.
20. Ling Z, Wei-li W, Shao-hua H. Forecast of incidence trend of influenza-like illness by the ARIMA model based on R. Zhonghua Yu Fang Yi Xue Za Zhi. 2018;22(9):957–60. https://doi.org/10.7629/yxdwfz202204010.
21. Shi X, Liu L, Shi Y, Zhao D, Zhang S. Analysis and prediction of influenza incidence in Shijiazhuang City by excel. J Med Pest Control. 2022;38(06):539–43. https://doi.org/10.7629/yxdwfz202206008.
22. Song X, Xiao J, Deng J, Kang Q, Zhang Y, Xu J. Time series analysis of influenza incidence in Chinese provinces from 2004 to 2011. Medicine (Baltimore). 2016;95(26). https://doi.org/10.1097/MD.0000000000003929.
23. Shumway RH, Stoffer DS, Shumway RH, Stoffer DS. ARIMA models. Time series analysis and its applications: with R examples. 2017:75–163. https://doi.org/10.1007/978-3-319-52452-8.
24. Ho SL, Xie M. The use of ARIMA models for reliability forecasting and analysis. Computers & industrial engineering. 1998;35(1–2):213–6. https://doi.org/10.1016/S0360-8352(98)00066-7.
25. Woodward WA, Gray HL, Elliott AC. Applied time series analysis with R. 2nd Edition ed. Boca Raton: CRC press; 2017. https://doi.org/10.1201/9781315161143.
26. Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; San Francisco: Association for Computing Machinery; 2016. p. 785–94. https://doi.org/10.1145/2939672.2939785.
27. Liu X, Li N, Liu S, Wang J, Zhang N, Zheng X, et al. Normalization methods for the analysis of unbalanced transcriptome data: a review. Front Bioeng Biotechnol. 2019;7:358. https://doi.org/10.3389/fbioe.2019.00358.
28. Kiranyaz S, Avci O, Abdeljaber O, Ince T, Gabbouj M, Inman DJ. 1D convolutional neural networks and applications: a survey. Mech Syst Signal Process. 2021;151:107398. https://doi.org/10.1016/j.ymssp.2020.107398.
29. Guessoum S, Belda S, Ferrandiz JM, Modiri S, Raut S, Dhar S, et al. The Short-Term Prediction of Length of Day Using 1D Convolutional Neural Networks (1D CNN). Sensors (Basel). 2022;22(23). https://doi.org/10.3390/s22239517.
30. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80. https://doi.org/10.1162/neco.1997.9.8.1735.
31. Lipton ZC, Berkowitz J, Elkan C. A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:150600019. 2015. https://doi.org/10.48550/arXiv.1506.00019.
32. Kim MH, Kim JH, Lee K, Gim G-Y. The prediction of COVID-19 using LSTM algorithms. Int J Networked Distrib Comput. 2021;9(1):19–24. https://doi.org/10.2991/ijndc.k.201218.003.
33. Muhammad LJ, Haruna AA, Sharif US, Mohammed MB. CNN-LSTM deep learning based forecasting model for COVID-19 infection cases in Nigeria, South Africa and Botswana. Health Technol (Berl). 2022;12(6):1259–76. https://doi.org/10.1007/s12553-022-00711-5.
34. Lu W, Li J, Li Y, Sun A, Wang J. A CNN-LSTM-Based model to forecast stock prices. Complexity. 2020;2020:6622927. https://doi.org/10.1155/2020/6622927.
35. Siami-Namini S, Tavakoli N, Namin AS, editors. A comparison of ARIMA and LSTM in forecasting time series. 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA); 2018 17–20 Dec. 2018.
36. Alim M, Ye GH, Guan P, Huang DS, Zhou BS, Wu W. Comparison of ARIMA model and XGBoost model for prediction of human brucellosis in mainland China: a time-series study. BMJ Open. 2020;10(12):e039676. https://doi.org/10.1136/bmjopen-2020-039676.
37. Luo J, Zhang Z, Fu Y, Rao F. Time series prediction of COVID-19 transmission in America using LSTM and XGBoost algorithms. Results Phys. 2021;27:104462. https://doi.org/10.1016/j.rinp.2021.104462.
38. Zhang J, Li S. Air quality index forecast in Beijing based on CNN-LSTM multi-model. Chemosphere. 2022;308(Pt 1):136180. https://doi.org/10.1016/j.chemosphere.2022.136180.
39. Yan R, Liao J, Yang J, Sun W, Nong M, Li F. Multi-hour and multi-site air quality index forecasting in Beijing using CNN, LSTM, CNN-LSTM, and spatiotemporal clustering. Expert Syst Appl. 2021;169:114513. https://doi.org/10.1016/j.eswa.2020.114513.
40. Hu CY, Tang YW, Su QM, Lei Y, Cui WS, Zhang YY, et al. Public health measures during the COVID-19 pandemic reduce the spread of other respiratory infectious diseases. Front Public Health. 2021;9:771638. https://doi.org/10.3389/fpubh.2021.771638.
41. Khorramdelazad H, Kazemi MH, Najafi A, Keykhaee M, Zolfaghari Emameh R, Falak R. Immunopathological similarities between COVID-19 and

influenza: investigating the consequences of co-infection. Microb Pathog. 2021;152:104554. https://doi.org/10.1016/j.micpath.2020.104554.
42. Zhao Z, Zhai M, Li G, Gao X, Song W, Wang X, et al. Study on the prediction effect of a combined model of SARIMA and LSTM based on SSA for influenza in Shanxi Province, China. BMC Infect Dis. 2023;23(1):71. https://doi.org/10.1186/s12879-023-08025-1.
43. Kim KH, Chang B, Choi HK. Deep learning based short-term electric load forecasting models using one-hot encoding. J IKEEE. 2019;23(3):852–7. https://doi.org/10.1016/j.engappai.2021.104645.
44. Athanasiou M, Fragkozidis G, Zarkogianni K, Nikita KS. Long short-term memory-based prediction of the spread of influenza-like illness leveraging surveillance, weather, and twitter data: model development and validation. J Med Internet Res. 2023;25:e42519.https://doi.org/10.2196/42519.
45. Wan Y, Song P, Liu J, Xu X, Lei X. A hybrid model for hand-foot-mouth disease prediction based on ARIMA-EEMD-LSTM. BMC Infect Dis. 2023;23(1):879. https://doi.org/10.1186/s12879-023-08864-y.
46. Morris M, Hayes P, Cox IJ, Lampos V. Neural network models for influenza forecasting with associated uncertainty using Web search activity trends. PLOS Comput Biol. 2023;19(8):e1011392. https://doi.org/10.1371/journal.pcbi.1011392.

Publisher's Note