RESEARCH ARTICLE

Deep Prediction Network Based on Covariance Intersection Fusion for Sensor Data

Hanchi Ren[1,*], **Yeqing Wang**[2] and **Huijun Ma**[2]

[1] Department of Computer Science, Swansea University, Swansea SA1 8EN, United Kingdom
[2] National Engineering Laboratory for Agri-product Quality Traceability, BTBU, Beijing, China

Abstract

To predict future trends based on the data from sensors is an important technology for many applications, such as the Internet of Things, smart cities, etc. Based on the predicted results, further decisions and system controls can be made. Raw sensor data sets are often complex non-linear data with noise, which results in the difficulty of accurate prediction. This paper proposes a distributed deep prediction network based on a covariance intersection (CI) fusion algorithm in which the deep learning networks, such as long short-term memory networks (LSTM) and gated recurrent unit networks (GRU) are fused by CI fusion algorithm to effectively improve the performance of prediction. Moreover, the variance is obtained to evaluate the prediction results. The model is validated on the real weather dataset in Beijing. The experiments show that LSTM and GRU have their pros and cons for different data, CI fusion can improve the accuracy of the final predictions, and the entire framework has robust prediction results with a reasonable estimated variance.

Keywords: deep prediction network, covariance intersection (CI) fusion, sensor data analytics.

1 Introduction

Time series data exists in many real-world systems, and the analysis and prediction of time series data can provide effective guidance for system control.

For the prediction of time series data, researchers have proposed some solutions. For example, the traditional Autoregressive Integrated Moving Average (ARIMA) model can only predict stationary time series and struggles with nonlinear data; hybrid ARIMA-neural network approaches have been proposed to address these limitations [1]. In paper [2], Thissen et al. proposed an SVM algorithm with certain modeling ability for nonlinear data, and SVMs have also been applied to financial time series forecasting [3], though parameter selection remains challenging in practice.

With the development of sensor technology, the information obtained is more and more abundant, and the data-driven deep learning model has been widely studied and applied. In particular, the recurrent neural network (RNN) [6] shows strong modeling capabilities in nonlinear time series data, and has also been widely adopted in sequence modeling tasks [4, 5]. However, RNNs suffer from vanishing and exploding gradient problems when capturing

Academic Editor:
Weimin Zhang

Submitted: 05 January 2024
Accepted: 19 May 2024
Published: 25 May 2024

Vol. 1, **No.** 1, 2024.
10.62762/TIS.2024.136898

***Corresponding author:**
Hanchi Ren
hanchi.ren@swansea.ac.uk

Citation
Ren, H., Wang, Y., & Ma, H. (2024). Deep Prediction Network Based on Covariance Intersection Fusion for Sensor Data. *ICCK Transactions on Intelligent Systematics*, 1(1), 10–18.

long-term dependencies. Improved RNNs such as LSTM [7–9] and GRU [10, 11] have been proposed and applied. LSTM sets three gating units to control the memory and forgetting of time series through cell state. LSTM solves the vanishing gradient and exploding gradient problem of RNN and realizes the long-term dependence of capturing data. GRU has reduced a gating unit based on LSTM, reducing network parameters while maintaining the modeling capabilities comparable to LSTM.

At the same time, bidirectional extensions of recurrent models improve sequence modeling [12], and BiLSTM in particular performs better in periodic time series [13], even if it needs to be trained for a longer time. In addition, in the case of multiple input variables, the RNN series network cannot be highly accurate because it cannot take into account the lateral relationship in multiple variables. In papers [14–16], authors combined the convolutional neural network CNN with LSTM, in which CNN extracts the local spatial features of multidimensional data, considering the horizontal relationship of multidimensional time series.

In addition, different network models have different performances on different data, and even if the neural network has been trained, the prediction variance in different time periods has fluctuations. Moreover, the output of different neural networks is difficult to judge by appropriate standards. Relying solely on the root mean square error is biased, as the values are often similar across models, making it difficult to accurately assess the reliability of the results. To this end, we can use multiple neural networks to model the data to form a distributed network, but this raises the problem of how to integrate these multiple prediction results effectively. Therefore, the use of effective fusion methods to synthesize the prediction results of different models can ensure the accuracy and stability of the output results. Commonly used fusion strategies include the weighted average method, direct averaging, and ensemble-based decomposition [20]. Graph neural network fusion has also been explored to jointly model spatial and temporal dependencies [17, 18]. However, weighted averaging requires careful weight determination, while nonlinear neural network fusion introduces additional parameters and cannot provide variance estimates for evaluating prediction reliability. In paper [21], the author uses the method of covariance intersection to fuse the features extracted by the neural network and improve the accuracy. The covariance intersection fusion algorithm can not only obtain interpretable suboptimal estimates in multi-source correlation data fusion but also give the variance of each estimated value to evaluate the correctness of the predicted values.

We designed a distributed deep learning network model combining traditional time series data decomposition technology and a fusion scheme. We use three models of LSTM, GRU, and ConvBiLSTM to perform multi-step prediction on the same time series variable respectively, and then use the covariance intersection fusion algorithm to fuse the prediction results of the three models to ensure the accuracy of the prediction results and give predictions with quantitative evaluation of the results. In implementing the variance required for CI fusion, the step estimation variance for each prediction time step is obtained by overlapping the prediction data. Finally, by estimating the variance of the step size, the covariance intersection algorithm is designed to fuse the prediction results, yielding more accurate predictions along with their possible fluctuation range.

Niehsen [22] verified that covariance intersection filtering provides a general framework for information fusion because it can produce consistent estimates for any degree of cross-correlation. In paper [21], covariance intersection algorithm was used in neural networks feature level fusion effectively improves the accuracy of the experiment. Hurley [23] extended covariance intersection to the fusion of any two probability density functions, and gave the minimum of the covariance matrix of the fusion, thus verifying the covariance intersection algorithm in multiple data. In paper [24], the authors compare the covariance intersection algorithm with other fusion algorithms. The interpretability of the CI is verified and the correct minimum estimated variance is given. Li and Nashashibi [19] applied the CI algorithm to the field of vehicle positioning. In multiple vehicle cooperative tasks, the different states of different vehicles were estimated and merged using the CI algorithm. In the case of unknown correlation, CI obtained the best agreement. It is estimated that there are clear advantages over other fusion methods. However, although the CI fusion method can obtain better results when the data source has an unknown correlation, the variance of the different data is required to be known in the application. For most sensor data, the variance can be estimated through measurement, and the variance of the output of the deep network model is not scientifically defined and cannot be used directly.

Based on the above analysis, we can know that in the face of real highly nonlinear data, neither the traditional mathematical equation model nor the data-driven deep network model can obtain the best results. Therefore, the introduction of appropriate time series decomposition techniques can effectively reduce the complexity of the original data, and in the case of insufficient data, the effective feature components are more conducive to the training of the model in the deep network model. However, neural networks are also not universal. Different time series variables and different time periods lead to variations in the prediction performance of neural networks. For this reason, in the face of the multi-step prediction problem of complex time series data, it is very meaningful to use the appropriate fusion method to synthesize the prediction results of different neural network models to obtain a more stable final prediction result. In the existing linear fusion method, there is a problem that the weight is difficult to determine, and the nonlinear ANN model will increase the complexity of the model. However, the correlation of the output of the neural network is unknown, so the CI fusion method becomes a natural choice. The research shows that the method still has suboptimal consistent estimation for data sources with unknown correlations. Faced with the problem that it is difficult to obtain the variance required by the CI algorithm. When we model the time series to predict certain values in the future, we can simultaneously model some known observations and unknown to-be-predicted values by enlarging the number of prediction steps, that is, in a prediction Multi-step, both obtaining forecasts of observations also includes predictions of future values. In this way, we can estimate the prediction estimation variance for the prediction period, and based on this variance, design the CI fusion algorithm to obtain more accurate results and quantitative evaluation.

Therefore, this paper proposes a distributed multi-step predictive deep network model based on CI fusion and uses LSTM, GRU, and ConvBiLSTM three models as sub-models. Among them, when using ConvBiLSTM, the decomposition method of STL is added to decompose the time series data into three feature components to improve the accuracy. Compared to the existing methods, we use different models for prediction and use CI fusion to obtain the final result after separately obtaining the prediction results. The CI fusion strategy also gives the variance of the predicted values at each moment, and the results are more reasonable and interpretable. This paper makes the following two contributions:

1. This paper establishes a general framework for the prediction of complex time series data, which combines the data-driven deep network model and CI fusion strategy to ensure the accuracy and quantitative evaluation of the prediction results.
2. For sensor data, it is often difficult to obtain the variance, so the prediction results of the data-driven deep network model are not easy to evaluate. Therefore, by predicting overlapping historical datapoints with an extended prediction window, we estimate the step-wise prediction variance for the CI fusion algorithm and based on this, a covariance fusion strategy is designed. Furthermore, the use of CI fusion variance to give a quantitative evaluation of the prediction results has a more important reference significance in practical applications.

The remainder of this paper is organized as follows: Section 2 introduces the proposed methodology. Section 3 presents the experimental validation on the Beijing weather dataset. Section 4 draws conclusions.

2 Methodology

We have built a generic prediction model, as shown in Figure 1. The three sub-predictors are trained in a supervised learning manner. The input and output of the sub-predictor are historical data from one step and future data from one step, respectively. We can see that the whole model consists of four main components: The three neural network submodels are GRU, LSTM, and ConvBiLSTM. Among them, when using the ConvBiLSTM model, due to the abstract feature extraction ability of the convolution operation, we use the seasonal and trend decomposition using loess (STL) decomposition method to decompose the original data into more efficient three-feature components, and train the predictor based on this data.

Recurrent neural networks (RNNs) maintain a memory based on historical contextual information, which makes them a natural choice for processing sequential data. Long Short-Term Memory networks introduce memory cells as information storage modules, enabling long-term memory of sequential data and resolving the vanishing and exploding gradient problems of RNNs.

LSTM uses a gating mechanism to enable the recurrent neural network to not only retain past information but

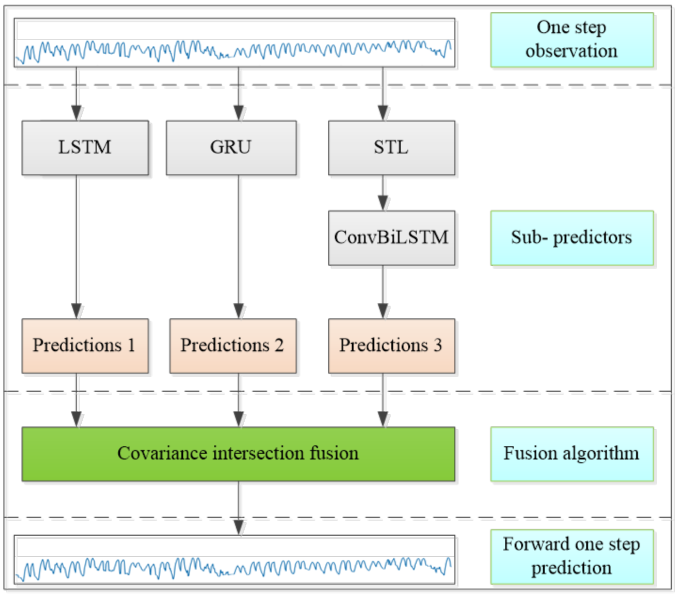


Figure 1. Model framework.

also selectively discard less important information to model long-term time dependencies. GRU is based on the idea of retaining long-term sequence information, which helps reduce the problem of vanishing gradient. The principle of GRU is very similar to that of LSTM, which uses gated mechanism to control input, memory and other information to make predictions at the current time step. The GRU has two gate units, a reset gate and an update gate. Intuitively, the reset gate determines how the new input information is combined with the previous memory, and the update gate defines the amount of the previous memory saved to the current time step.

Using the three sub-models to strongly model the nonlinear data ensures the prediction performance of completely different data sources. Finally, using the covariance intersection algorithm for three data sources, the prediction results of the three sub-models are combined to further improve accuracy, at the same time, the covariance intersection algorithm will give a possible range of fluctuations as a quantitative evaluation of the prediction results.

In the CI fusion algorithm, we add some time sensor data obtained to the one-step forward prediction, compare the predicted result of the obtained data with the true value of the moment, and estimate the overall variance of the prediction result in the previous step. Based on the variance, the CI algorithm is designed to fuse the results. Our data format is shown in Figure 2.

In the target tracking, in order to avoid errors caused

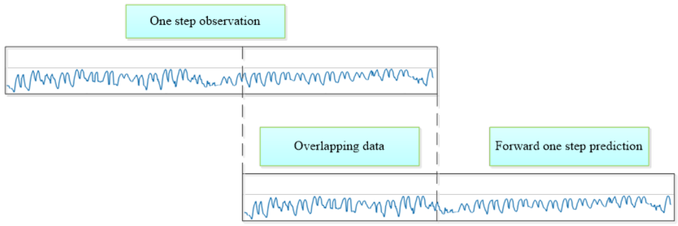


Figure 2. Data format.

by the influence of information redundancy in the Kalman filter estimation, the covariance information must be maintained, but in a fully distributed system, the cross-covariance information cannot be uniformly maintained. In order to solve the above problems, Julier and Uhlmann [24] proposed a data fusion mechanism that does not require independence assumptions, which can be applied to the covariance intersection fusion algorithm of arbitrary complex distributed systems. We designed a covariance intersection algorithm that uses three sources of information. A general covariance algorithm can fuse multiple sources of information. Suppose multiple information source data is y^m, m represents the number of information sources. The intersection represents the convex combination of covariance, and the covariance intersection algorithm is as follows:

$$P^{-1} = \omega_1 P_1^{-1} + \omega_2 P_2^{-1} + \ldots + \omega_m P_m^{-1} \quad (1)$$

$$\begin{aligned} & P^{-1}y = \omega_1 P_1^{-1} y_1 + \omega_2 P_2^{-1} y_2 + \cdots + \omega_m P_m^{-1} y_m \\ & \text{s.t.} \quad 1)\ \omega_1, \omega_2, \ldots, \omega_m \in [0,1] \\ & \qquad\ \ 2)\ \omega_1 + \omega_2 + \cdots + \omega_m = 1 \end{aligned} \quad (2)$$

where ω_m represents the weight of the m data source, P_m represents the variance of the m data source, y represents the fusion result. By applying a convex optimization algorithm to determine the appropriate weights, the optimal fusion result can be obtained.

In our paper, three sub-predictors were chosen, so there are three data sources. The estimated variance of the three sub-predictors is obtained by prediction of repeated data of adjacent step sizes. The values at the same time are predicted by three sub-predictors, and the results are $\hat{y}_n^1$, $\hat{y}_n^2$, $\hat{y}_n^3$, respectively. Where n represents the moment. The estimated variance is P_1, P_2, P_3 which are calculated by formula (3) respectively and covariance P is calculated by formula (4).

$$\hat{P}_i = \sqrt{\frac{1}{13} \sum_{r=1}^{13} \left(\hat{y}_{24}^{i,r} - \hat{y}_{37}^{r} \right)^2} \quad (3)$$

$$P^{-1} = \omega_1 P_1^{-1} + \omega_2 P_2^{-1} + \omega_3 P_3^{-1} \tag{4}$$

where ω_1, ω_2, ω_3 represent the weights assigned to the predicted values of each sub-model, respectively

$$\begin{aligned} \min_{\omega_1,\omega_2,\omega_3} \quad & P^{-1} = \omega_1 P_1^{-1} + \omega_2 P_2^{-1} + \omega_3 P_3^{-1} \\ \text{s.t.} \quad & 1)\ \omega_1, \omega_2, \omega_3 \in [0,1] \\ & 2)\ \omega_1 + \omega_2 + \omega_3 = 1 \end{aligned} \tag{5}$$

We use the SLSQP algorithm [25] to optimize P^{-1} under the constraints on ω_1, ω_2, and ω_3 in formula (5). Then, calculate the result of the fusion according to formula (6).

$$P^{-1}\hat{y}_n = \omega_1 P_1^{-1}\hat{y}_n^1 + \omega_2 P_2^{-1}\hat{y}_n^2 + \omega_3 P_3^{-1}\hat{y}_n^3 \tag{6}$$

where $\hat{y}_n$ represents the fusion result at time n.

The flow of our entire algorithm is as follows pseudo code, where the symbol convention is as follows: the agreed symbols of t, T are set to 24, 37 respectively, representing the forward 24 step prediction. y_1 to y_{37} represent 37 real values of historical moments. $\hat{y}_{24}$ to $\hat{y}_{37}$ represent 13 predicted values of historical moments and $\hat{y}_{38}$ to $\hat{y}_{61}$ represent 24 forward predicted values.

3 Experiments

3.1 Dataset

The data in our experiments come from the meteorological dataset used in a Global AI Challenge contest in 2018, which focuses on real-world meteorological data observed at a weather station in Beijing, including meteorological factors such as temperature, relative humidity, and wind speed. The data set has high continuity with fewer missing values. Our experiments are based on two variables, temperature and wind speed. These two variables exhibit distinct trends: temperature changes show clear periodicity, whereas wind speed exhibits abrupt fluctuations. Data is collected every hour. We selected continuous 200-day data as the data source and filled in the missing values with data from the immediately preceding time point. We use 170 days of data as a training set for network models, and the remaining 30 days of data as a test set.

3.2 Experiment setup

The experiment hardware and software environments are set up to run the proposed prediction model. The open source deep learning library Keras, based on TensorFlow, is used to build all learning models. All experiments are performed on a PC with an Intel(R) CORE(TM) CPU i5-4200U 1.60 GHz and 4 GB of memory.

To train the deep neural networks effectively, a number of hyperparameters must be configured. In experiments, the default parameters in Keras are used for deep neural network initialization such as weight initialization and Learning rate. Usually, when we use neural networks to build models, the number of layers and neurons are not strictly predefined; instead, the model complexity is determined empirically based on the data. We determine the parameters of each layer of the model through multiple experimental adjustments. In addition, the Tanh activation function is used for the LSTM, GRU, and Dense layers of the MLP fusion baseline. The activation function of the convolutional layers is set to ReLU. The size and hyper parameter details of each network model are shown in Table 1.

All models are optimized using the Adam optimizer with default Keras settings ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-7}$, learning rate $= 0.001$). No dropout regularization is applied. The dataset is split into a training set (170 days), a validation set (the last 10 days of the training period, used for monitoring convergence), and a test set (30 days). No data augmentation is performed. All experiments are repeated five times with different random seeds, and the mean RMSE is reported.

3.3 Hyperparameter Selection Strategy

The hyperparameters of each sub-model are determined through multiple experimental adjustments. The batch size is set to 20 for all recurrent models and 30 for the BP network, with the number of training epochs set to 4000 and 100, respectively, as detailed in Table 1. The default parameter settings in Keras are adopted for weight initialization and learning rate. The activation functions are set to Tanh for LSTM, GRU, and BP layers, and ReLU for the convolutional layers in ConvBiLSTM. For the CI fusion module, the weights ω_1, ω_2, ω_3 are optimized using Sequential Least Squares Programming (SLSQP) at each prediction step, ensuring that the fused result satisfies the convexity constraints defined in formula (5).

Table 1. Hyperparameters details for all experiments.

Model	Number of layers	size	Experiment setup
GRU	2 GRU	{37}, {37}	Batch size: 20 Epochs: 4000
LSTM	2 LSTM	{37}, {37}	Batch size: 20 Epochs: 4000
ConvBiLSTM	2 CNN & 1 BiLSTM &1 LSTM & 1 Dense	{3,3}, {3,3}, {37}, {37}, {37}	Batch size: 20 Epochs: 4000
BP	3 Dense	{3}, {5}, {1}	Batch size: 30 Epochs: 100

A detailed introduction of the three sub-predictors is as follows.

1. GRU: In this model, the raw data was not processed except for data preprocessing-related operations, and it was used to train the GRU network to build a predictive model.
2. LSTM: Train the network with exactly the same data as the submodel of GRU.
3. ConvBiLSTM: Using the same data as the previous two models as the data source, the three feature components are obtained through the STL decomposition method before the data enters the network. Use this feature component to train this network to obtain a predictor.

The method proposed in this paper mainly used the fusion idea of distributed network, obtains the prediction results of the same time period through three sub-models, and designs the covariance intersection fusion algorithm at the end, so as to synthesize the results of the three models and obtain the final predictions. The prediction performance of different models was evaluated by comparison with real values. The root mean square error (RMSE) was used to estimate the performance of models. RMSE is frequently used to measure the difference between values predicted by a model and the values actually observed from the environment. A value of 0 indicates that the observed value exactly fits the predicted value. The calculation is as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{7}$$

where $\hat{y}_i$ represents the predicted value, y_i represents the truth value, and n represents the number of test data.

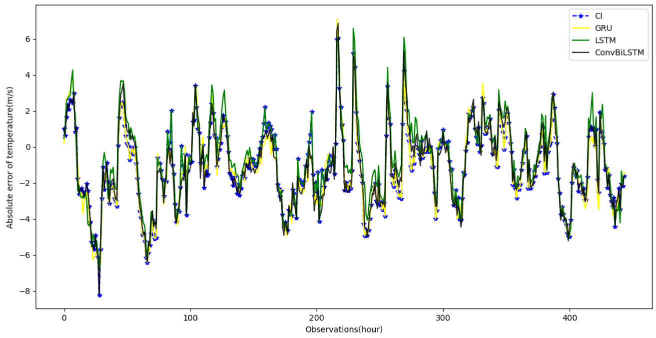
To thoroughly evaluate the proposed method, two sets of experiments were conducted. The first set focuses on validating the effectiveness of the CI fusion algorithm by comparing its performance with the three individual sub-predictors (LSTM, GRU, and ConvBiLSTM) on temperature and wind speed prediction. The results indicate that the CI fusion approach achieves superior prediction accuracy compared to the single models.

The second set of experiments compares the CI fusion algorithm with the back-propagation (BP) neural network fusion method. The results demonstrate that the CI fusion algorithm not only provides higher prediction accuracy but also delivers a reasonable fluctuation range via variance estimation, making it more practical for real-world sensor data applications.

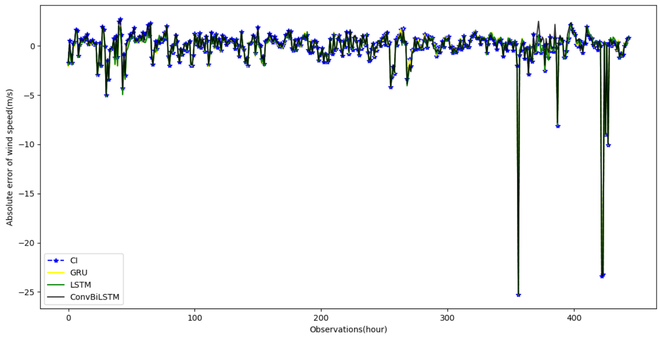
3.4 Results of Case No. 1

We completed two experiments on a real dataset, each of which included predictions of two meteorological variables, temperature (T) and wind speed (WS). We process the data into pairs of data with overlapping moments, recording 37 time points per day, of which 13 are historical observations and 24 are observations for the following day. We use the data from the previous day to predict the trend of the next day, thus building a forward 24 step prediction model. Results of the first experiment are shown in Figure 3(a) and Figure 3(b). The prediction results of T and WS are shown in Figure 4(a) and Figure 4(b) respectively. The red line presents the ground truth of T and WS, and the blue line presents fusion results by CI. The green, yellow, and black lines are the predictive results with LSTM, GRU and ConvBiLSTM models, respectively.

Figure 3(a) compares the ground truth (real) data with the 24-step forward predictions of two non-stationary time series generated by four models. The results indicate that a single GRU or LSTM model is not suitable for predicting non-stationary time series data. This is because the trend of different variables is often very different, for example, the temperature has obvious periodicity, and the wind speed often has sharp peaks. Although the ConvBiLSTM model combines CNN and BiLSTM to extract features of local spatial dimension and temporal dimension, there is no



(a) Temperature predictions absolute error.



(b) Wind speed predictions absolute error.

Figure 3. Absolute prediction errors of LSTM, GRU, ConvBiLSTM, and CI fusion.

ideal accuracy on the real dataset. The main reason is that the ConvBiLSTM network is relatively large with more parameters, and it is easy to overfit on small data sets.

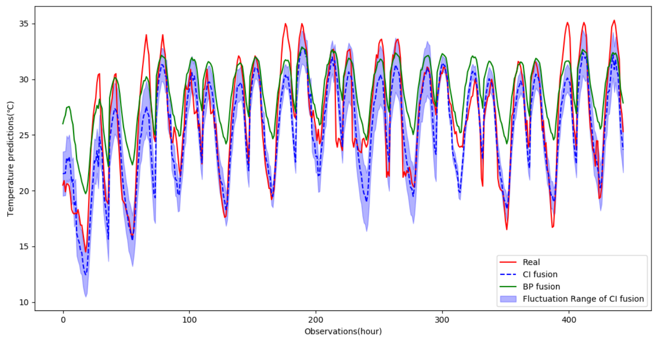
Figure 3 shows the absolute error of the predictive results and ground truth of non-stationary time series. The blue line represents the fusion results by CI and the green, yellow, black lines are the predictive results with LSTM, GRU and ConvBiLSTM models, respectively. The closer the difference is to 0, the more accurate the predictions. The quantitative results are shown in Table 2, alongside RMSE comparative analysis of four models.

Table 2. RMSE comparison of different models for temperature (T) and wind speed (WS).

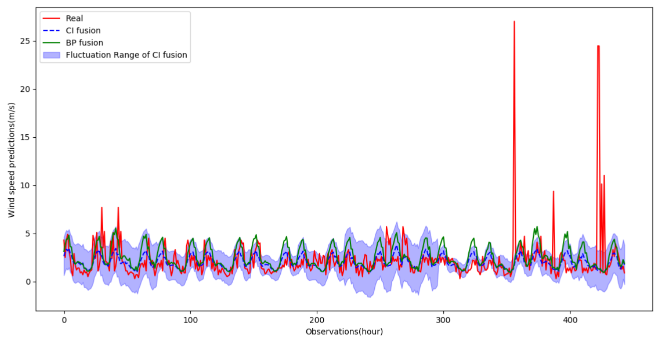
Model	RMSE(T)	RMSE(WS)
LSTM	2.4389	2.5102
GRU	2.4612	2.5234
ConvBiLSTM	2.4131	2.4843
CI fusion	2.3235	2.4784

As shown in Table 2, the CI fusion method achieves better performance than the ConvBiLSTM model. It reduces the RMSE by approximately 3.71% for temperature prediction and by 0.24% for wind speed prediction. The primary reason for this is that the proposed CI based model can give an improved predicted value even if the correlation between the predicted value and the true value is unknown. At the same time, the CI algorithm comprehensively considers the predicted value and variance information of each data source when merging multiple data sources, thereby increasing the reliability of the results. In addition, in the prediction of wind speed data, the results of the CI fusion method are also slightly improved. This is mainly because the sampling interval of the wind speed is 1 hour and the data is extremely abrupt. Therefore, in multi-step prediction, it is difficult to correctly predict the peak point. Even so, the CI fusion model gives stable and better results than a single predictor such as LSTM and GRU.

3.5 Results of Case No. 2



(a) Fusion results of T predictions.



(b) Fusion results of WS predictions.

Figure 4. Prediction results of CI fusion and BP fusion with estimated fluctuation range.

In the second experiment, we mainly compared the results of the two fusion methods, namely the linear CI fusion method and the nonlinear back-propagation (BP) neural network fusion method. As described in Section 2, the CI fusion method gives an optimal variance of the results, while the BP fusion algorithm can only give a series of predicted values. Figure 4 shows the prediction results of the two fusion methods. Based on this, we propose a more reasonable evaluation method, which uses the optimal variance of the predicted values to estimate the possible fluctuation range of the predicted values. As can be seen from Figure 4, most of the values in the predicted sequence are within reasonable fluctuations. A few outliers can be understood as noise of data.

The RMSE values of BP fusion and CI fusion are shown in Table 3. The CI fusion algorithm outperforms the BP fusion method, reducing the RMSE by 10.10% for temperature and by 0.47% for wind speed prediction.

Table 3. RMSE comparison between BP fusion and CI fusion for temperature (T) and wind speed (WS).

Model	RMSE(T)	RMSE(WS)
BP fusion	2.5847	2.4901
CI fusion	2.3235	2.4784

4 Conclusion

This paper mainly establishes a general multi-step prediction network framework for complex sensor time series data. Firstly, three sub-models — LSTM, GRU, and ConvBiLSTM — are used to model the same data synchronously. In the sub-predictor ConvBiLSTM, the original data is decomposed into simple sub-sequences by using STL decomposition technology to reduce the influence of noise on network training, and then the combined convolution and BiLSTM network is established to extract the features on the two dimensions of the horizontal and time stamps for the multi-feature components. By designing the CI fusion algorithm, the results of the three sub-models are combined to achieve higher prediction accuracy. This not only ensures the accuracy of different variables, but also gives a reasonable quantitative evaluation method. The three sub-models have different performances in different variables and different prediction periods. For this reason, we use CI methods to synthesize different prediction results to ensure the accuracy of the framework. In the CI fusion module, we calculate the step size estimation variance by overlapping data, and finally obtain more accurate prediction results and range of variation, which provides a more comprehensive evaluation of the prediction results beyond point estimates alone.

Data Availability Statement

Data will be made available on request.

Funding

This work was supported by the National Natural Science Foundation of China under Grant 62173002.

Conflicts of Interest

Hanchi Ren served as an Associate Editor of *ICCK Transactions on Intelligent Systematics* at the time of manuscript submission. To ensure the integrity of the peer-review process, Hanchi Ren was not involved in the editorial handling, peer review, or decision-making process for this manuscript, which was handled independently by another editor. The remaining authors declare no conflicts of interest.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, *50*, 159-175. [CrossRef]

[2] Thissen, U. V. B. R., Van Brakel, R., De Weijer, A. P., Melssen, W. J., & Buydens, L. M. C. (2003). Using support vector machines for time series prediction. *Chemometrics and intelligent laboratory systems*, *69*(1-2), 35-49.[CrossRef]

[3] Tay, F. E., & Cao, L. (2001). Application of support vector machines in financial time series forecasting. *omega*, *29*(4), 309-317. [CrossRef]

[4] Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., & Khudanpur, S. (2010, September). Recurrent neural network based language model. In *Interspeech* (Vol. 2, No. 3, pp. 1045-1048). [CrossRef]

[5] Mikolov, T., Kombrink, S., Burget, L., Černocký, J., & Khudanpur, S. (2011, May). Extensions of recurrent neural network language model. In *2011 IEEE international conference on acoustics, speech and signal processing* (*ICASSP*) (pp. 5528-5531). IEEE.[CrossRef]

[6] Connor, J. T., Martin, R. D., & Atlas, L. E. (1994). Recurrent neural networks and robust time series prediction. *IEEE transactions on neural networks*, *5*(2), 240-254. [CrossRef]

[7] Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. *Neural computation*, *12*(10), 2451-2471.[CrossRef]

[8] Fu, R., Zhang, Z., & Li, L. (2016, November). Using LSTM and GRU neural network methods for traffic flow prediction. In *2016 31st Youth academic annual conference of Chinese association of automation* (*YAC*) (pp. 324-328). IEEE.[CrossRef]

[9] Malhotra, P., Vig, L., Shroff, G., & Agarwal, P. (2015). Long short term memory networks for anomaly detection in time series. In *ESANN 2015: Proceedings of the 23rd European Symposium on Artificial Neural Networks* (pp. 89-94).

[10] Che, Z., Purushotham, S., Cho, K., Sontag, D., & Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Scientific reports, 8*(1), 6085. [CrossRef]

[11] Yao, S., Hu, S., Zhao, Y., Zhang, A., & Abdelzaher, T. (2017, April). Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th international conference on world wide web* (pp. 351-360). [CrossRef]

[12] Osogami, T., Kajino, H., & Sekiyama, T. (2017, July). Bidirectional learning for time-series models with hidden units. In *International Conference on Machine Learning* (pp. 2711-2720). PMLR. [CrossRef]

[13] Siami-Namini, S., Tavakoli, N., & Namin, A. S. (2019, December). The performance of LSTM and BiLSTM in forecasting time series. In *2019 IEEE International conference on big data (Big Data)* (pp. 3285-3292). IEEE. [CrossRef]

[14] Liu, Y., Zheng, H., Feng, X., & Chen, Z. (2017, October). Short-term traffic flow prediction with Conv-LSTM. In *2017 9th international conference on wireless communications and signal processing (WCSP)* (pp. 1-6). IEEE.[CrossRef]

[15] Du, S., Li, T., Yang, Y., & Horng, S. J. (2019). Deep air quality forecasting using hybrid deep learning framework. *IEEE Transactions on Knowledge and Data Engineering, 33*(6), 2412-2424. [CrossRef]

[16] Kim, S., Hong, S., Joh, M., & Song, S. K. (2017). Deeprain: Convlstm network for precipitation prediction using multichannel radar data. *arXiv preprint arXiv:1711.02316*.

[17] Chen, Y., & Xie, Z. (2022). Multi-channel fusion graph neural network for multivariate time series forecasting. *Journal of Computational Science, 64*, 101862. [CrossRef]

[18] Li, M., & Zhu, Z. (2021, May). Spatial-temporal fusion graph neural networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 35, No. 5, pp. 4189-4196).

[19] Li, H., & Nashashibi, F. (2013). Cooperative multi-vehicle localization using split covariance intersection filter. *IEEE Intelligent transportation systems magazine, 5*(2), 33-44. [CrossRef]

[20] Qiu, X., Ren, Y., Suganthan, P. N., & Amaratunga, G. A. (2017). Empirical mode decomposition based ensemble deep learning for load demand time series forecasting. *Applied soft computing, 54*, 246-255. [CrossRef]

[21] Pu, L., Feng, X., & Hou, Z. (2019, May). Covariance intersection fusion for visual tracking with hierarchical features. In *Tenth International Conference on Graphics and Image Processing* (ICGIP 2018) (Vol. 11069, pp. 1158-1166). SPIE.[CrossRef]

[22] Niehsen, W. (2002, July). Information fusion based on fast covariance intersection filtering. In *Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002.(IEEE Cat. No. 02EX5997)* (Vol. 2, pp. 901-904). IEEE.[CrossRef]

[23] Hurley, M. B. (2002, July). An information theoretic justification for covariance intersection and its generalization. In *Proceedings of the fifth international conference on Information Fusion. FUSION 2002.(IEEE Cat. No. 02EX5997)* (Vol. 1, pp. 505-511). IEEE. [CrossRef]

[24] Julier, S., & Uhlmann, J. K. (2017). General decentralized data fusion with covariance intersection. In *Handbook of multisensor data fusion* (pp. 339-364). CRC Press.

[25] Ma, Y., Zhang, N., & Li, J. (2022). Improved sequential least squares programming–driven feasible path algorithm for process optimisation. In *Computer aided chemical engineering* (Vol. 51, pp. 1279-1284). Elsevier. [CrossRef]

Hanchi Ren graduated from Swansea University in 2023 with a PhD's degree in Computer Science. His research focuses on Computer Vision, Privacy Preservation, Machine Learning, Deep Learning. (Email: hanchi.ren@swansea.ac.uk)

Yeqing Wang graduated from Beijing Technology and Business University in 2022 with a Bachelor's degree in Automation. He is currently a graduate student in Control Theory and Control Engineering at the same university. His research interests include pattern recognition and prediction, deep Learning, and related fields. (Email: wangyeqing@st.btbu.edu.cn)

Huijun Ma graduated from Changchun Institute of Optics and Mechanics with a master's degree in atomic and molecular physics in 2010. She is currently an on-the-job doctoral student in systems science at Beijing Technology and Business University. Research directions include complex system modeling, pattern recognition and information fusion, machine learning, etc. (Email: mahuijun@th.btbu.edu.cn)