Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8280428, 10 pages
https://doi.org/10.1155/2022/8280428

Research Article

An Improved Kalman Filter Based on Long Short-Memory Recurrent Neural Network for Nonlinear Radar Target Tracking

Fei Song,[1,2] **Yong Li**,[1] **Wei Cheng**,[1] **Limeng Dong**,[1] **Minqi Li**,[3] **and Junfang Li**[2]

[1]*School of Electronic Information, Northwestern Polytechnical University, Xi'an 710072, China*
[2]*School of Electronic Engineering, Xi'an Aeronautical Institute, Xi'an 710077, China*
[3]*School of Electronic Information, Xi'an Polytechnic University, Xi'an 710048, China*

Correspondence should be addressed to Yong Li; ruikel@nwpu.edu.cn

Received 26 May 2022; Accepted 25 July 2022; Published 21 August 2022

Academic Editor: A.H. Alamoodi

The target tracking of nonlinear maneuvering radar in dense clutter environments is still an important but difficult problem to be solved effectively. Traditional solutions often rely on motion models and prior distributions. This paper presents a novel improved architecture of Kalman filter based on a recursive neural network, which combines the sequence learning of recurrent neural networks with the precise prediction of Kalman filter in an end-to-end manner. We employ three LSTM networks to model nonlinear motion equation, motion noise, and measurement noise, respectively, and learn their long-term dependence from a large amount of training data. They are then applied to the prediction and update process of Kalman filter to calculate the estimated target state. Our approach is able to address the tracking problem of nonlinear maneuvering radar target online end-to-end and does not require the motion models and prior distributions. Experimental results show that our method is more effective and faster than the traditional methods and more accurate than the method with LSTM network alone.

1. Introduction

Target tracking is an important support for radar, sonar, satellite, optical sensor, and other systems to realize monitoring, positioning, navigation, and other applications. Its working mechanism is to fuse the prior information of the target and the online measurement information of the sensor and estimate the number, position, and motion state of the target online under the background of noise. In general, there are two key difficulties in target tracking, namely, target dynamic state uncertainty and measurement data uncertainty. This paper only focuses on the dynamic state uncertainty in radar target tracking.

For the single target tracking of approximately uniform motion, it is basically solved based on Bayesian theory. The essence of Bayesian theory is to seek the solution of the posterior probability density of motion state according to the prior probability density and observation likelihood function of dynamic parameters. Kalman filter (KF) [1] is an unbiased optimal estimation in the recursive form of Bayesian theory, which is applicable to target tracking of linear Gaussian motion in many fields. For nonlinear object tracking, the common improvement method is the Extended Kalman filter (EKF) [2], which approximately linearizes the system near the working point. The Unsensitive Kalman filter (UKF) [3] makes an approximate Gaussian distribution after projection of a few deterministic sample points. The Particle filtering (PF) [4] approximates the posterior probability density of the nonlinear functions by a large number of random discrete samples. These methods all approximate the posterior probability distribution from different directions. And their computational accuracy is limited by the model approximation effectiveness, or they need to balance the effectiveness and computational efficiency.

For target tracking with strong maneuverability, target maneuvering leads to the change of target dynamic characteristics. The commonly used solutions are the Markov jump multimodel algorithms, e.g., the interactive multimodel (IMM) algorithm and the variable structure multimodel algorithm (VSMM) algorithm [5, 6]. In essence, this kind of algorithms applies multiple motion models for matching the motion patterns of maneuvering target, with the models

transferred by Markov matrix. The disadvantage is that the model structure and quantity need to be determined in advance, leading to limited statistical accuracy due to model mismatch and insufficient estimation data.

For the nonlinear maneuvering radar target, its motion equation is often unpredictable, and the distribution of the motion process noise and measurement noise is also uncertain. These problems are difficult to solve using the traditional methods mentioned above. We analyze, in essence, the target tracking is to estimate the nonlinear mapping from the observation sequence to the real state of the target in time dimension. Considering the rapid development of deep neural networks (DNN) [7–11] in recent years, relying on its strong expression ability, it is possible to find a deterministic map to approximate the conditional density from a given input to an output in a mathematically optimal way by learning from a large amount of data [12]. Recurrent neural network (RNN) plays an important role in processing time series as a vital branch of DNN. Its main specialities are able to process historical data cyclically and recursively and model the historical memory information. With this help, RNN is good at dealing with the time series, which have tight correlation sequence information and uncertain length. Theoretically, as long as there is enough training data covering the possible motion path of the tracked object and an appropriate network structure, RNN is able to output the target state condition density of a given available observation value at each time step [13]. This coincides with the needs of target tracking. Inspired by this, we integrate the deep neural network with the traditional target tracking method, giving full play to their advantages, so as to effectively solve the above complex target tracking problems.

The contributions of this paper are summarized in the following items:

(1) In order to overcome the problems of nonlinear maneuvering radar target tracking, inspired by deep learning ideas, we propose a novel target tracking architecture combing Kalman filter and LSTM network. We model three LSTM networks separately for the uncertainty of the nonlinear motion equation, motion process noise distribution, and measurement noise distribution in Kalman filter. Relying on the powerful learning ability of LSTM network, the long-term dependent characteristics of target motion and measurement can be learned from the training of abundant training data and uniformly applied to the motion state prediction of Kalman filter. The advantages of this architecture are that it requires no approximation and no prior distribution, and it can carry out the nonlinear maneuvering radar target tracking end-to-end online

(2) We generate a large amount of training data using model sampling, based on widely used nonlinear maneuvering radar target time series model

(3) The qualitative and quantitative simulation results show that the proposed architecture is stronger than the traditional nonlinear target tracking methods in both the estimation error and computational speed in nonlinear maneuvering target tracking task

2. Related Work

In recent literatures, the application of DNN network in target estimation and prediction has been widely studied. In this section, we introduce those works that explicitly use LSTM in combination with traditional methods to infer time correlation in chronological order.

Early, Haarnoja et al. [14] used the one-time estimation as the measured value inputting into the Kalman filter. And the estimator is required to provide the prediction of noise covariance, which is often inaccurate without a learning process. Coskun et al. [15] proposed a method to human image pose regularization, which uses the LSTM estimator fused with a KF. However, it is only applicable to the estimation of slowly changing human image posture, and it does not fully explain the use performance of target tracking problem with rapidly changing state. Recently, Gao et al. [16] proposed two LSTM networks to receive the observation results and output the real state in a continuous manner. They complete the process of prediction and filtering, respectively. However, the two processes are independent and not related. The accuracy of the estimate will be limited. Moreover, they use an overall LSTM structure for estimation, which makes it more difficult to fit. Llerena et al. [17] used the LSTM unit to estimate the motion state through the encoder decoder architecture. However, only the performance comparison with EKF algorithm, and only under several linear and specific motion paths, cannot fully explain the superiority of this algorithm. Zhang et al. [18] used a Kalman filter based on LSTM for data assimilation of a two-dimensional spatiotemporal depth average ocean flow field and path planning of underwater glider. The LSTM network only models the spatial basis function in the nontidal flow field. The output of LSTM and the observation results obtained from the glider flow estimation data both are inputted to the Kalman filter. LSTM does not estimate all motion equations, process noise, and measurement noise, so the accuracy of estimation effect is limited.

In [19], we proposed to use of the LSTM network commonly used in RNN to extract the features of continuous data. Thus, it is able to fit the nonlinear mapping between noisy observations and target state under the conventional motion model. However, for nonlinear maneuvering radar target tracking, the ability of this method is limited. Therefore, we propose a new architecture combining LSTM network and Kalman filter, which strengthens the extraction of target motion history features, prediction accuracy, and noise resistance.

Our architecture is somewhat similar to [16], but is different in three key ways: First, motion state prediction and filtering in [16] are independently completed, while we integrate the results of the LSTM prediction into the Kalman filter state prediction to enhance the accuracy of the target prediction. Secondly, a whole RNN is used in [16] for learning and estimation, while we use three LSTMs: motion state, motion process noise, and measurement noise for learning and estimation. In this way, the learned historical information is more comprehensive and has stronger anti-interference ability. Thirdly, due to the different network architecture, we also use different loss functions.

3. Background

The Bayesian principle is applied as a basic theoretical framework for dealing with the observation data and its sequential characteristics uncertainty. The Bayesian method uses the prior estimation of motion state and the likelihood function of observation value to accurately estimate the motion state. We assume the motion model of the system is as follows:

$$\begin{aligned} \text{State equation} &: x_k = f(x_{k-1}) + v_k, \\ \text{Observation equation} &: y_k = h(x_k) + w_k, \end{aligned} \tag{1}$$

where x_k is the motion state at k step, v_k is the prediction process noise, which obeys the Gaussian distribution $\mathrm{v}_k \sim N(0, \mathrm{Q})$, and $f()$ is the motion equation, y_k is the observation data at the k step, w_k is the observation noise, which obeys the Gaussian distribution $w_k \sim N(0, R)$, and h is the observation equation.

The best solution of Bayesian filtering on the math from the classical Bayesian theory is the following:

$$p(x_k|y_{k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|y_{k-1})dx_{k-1}, \tag{2}$$

$$p(x_k|y_k) = \frac{1}{p(y_k|y_{k-1})} p(y_k|x_k)p(x_k|y_{k-1}). \tag{3}$$

From these, the Bayesian filtering consists chiefly of two stages: prediction stage as Equation (2) and filtering stage as Equation (3). We are able to accurately acquire the posterior probability density of the two stages. The Kalman filter algorithm is the recursive form to realize this accurate solution.

3.1. Kalman Filter. If we use x_t to represent the motion state and y_t to represent the measurement, we can change the model to

$$x_t = Ax_{t-1} + w \; w \sim N(0, Q), \tag{4}$$

$$y_t = Hx_t + v \quad v \sim N(0, R), \tag{5}$$

where A is the motion matrix, $x_{\mathrm{t}\text{-}1}$ is the motion state at time $t-1$, and w is process noise, which is a Gaussian white noise with zero mean and covariance matrix Q. H is the observation matrix, and v is measurement noise, which is a Gaussian white noise with zero mean and covariance matrix R.

The Kalman filter algorithm is divided into two stages: prediction and update.

The prediction stage calculates the state prior estimation and the error covariance prior estimation at current time according to the state estimate at previous time, as follows:

$$\widehat{x}_t' = A\widehat{x}_{t-1}, \tag{6}$$

$$\widehat{P}_t' = A\widehat{P}_{t-1}A^T + Q. \tag{7}$$

The update stage combines the prior estimations with new measurement data to calculate the improved posteriori estimations. K_t is the calculated filter gain matrix and the intermediate calculation result of filtering. The Kalman filter algorithm is a recursive prediction update method.

$$K_t = \widehat{P}_t'H^T\left(H\widehat{P}_t'H^T + R\right)^{-1}, \tag{8}$$

$$\widehat{x}_t = \widehat{x}_t' + K_t\left(\widehat{y}_t - H\widehat{x}_t'\right), \tag{9}$$

$$\widehat{P}_t = (I - K_T H)\widehat{P}_t'. \tag{10}$$

3.2. Long Short-Term Memory. With the help of RNN's characteristics as mentioned above, RNN is best adapted for dynamic scenes, hardly characterized by a fixed number of parameters. It is adequate for handling strongly correlated information in time and space series such as target tracking.

However, the traditional RNN is bad at storing the long-term memory of sequence information and may cause serious gradient disappearance or gradient explosion [20]. The main method to solve this problem is to use long-term memory network (LSTM), which is able to overcome the above shortcomings effectively.

LSTM is an improved RNN adding special memory function. It is able to compute the state of hidden layer neurons in various ways. The memory unit is mainly composed of four components: input gate, forgotten gate, output gate, and self-circulation connection. The LSTM protects and controls the status of the memory units by dominating the gates outputs. They work together to make LSTM get the ability to store and transmit sequence information for a long time and then reducing the problem of gradient disappearance.

In this paper, we use a general variant of forgetting gate [21], which is good for improving performance [22]. The LSTM variants are described below:

$$f_t = \sigma\left(W_{fh}h_{t-1} + W_{fx}x_t + b_f\right), \tag{11}$$

$$i_t = \sigma(W_{ih}h_{t-1} + W_{ix}x_t + b_i), \tag{12}$$

$$o_t = \sigma(W_{oh}h_{t-1} + W_{ox}x_t + b_o), \tag{13}$$

$$\tilde{c}_t = \tanh\left(W_{ch}h_{t-1} + W_{cx}x_t + b_c\right), \tag{14}$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes \tilde{c}_t, \tag{15}$$

$$h_t = o_t \otimes \tanh\left(c_t\right), \tag{16}$$

where σ represents sigmoid function and $\bigotimes$ represents element level multiplication. From these equations, it can be seen that the LSTM applies the forgotten gate f_t to reset the memory, uses the input gate i_t to write to the memory, employs the output gate o_t to read from the memory, and finally forms the output or hidden state h_t. The values of the intermediate storage unit $\tilde{c}_t$ and all gates are determined by the input x_t and all kinds of W and b parameters, which are learned in a training process. For multilayer LSTM, the first layer hidden states h_t are regarded as the input x_t of the second layer, and so on.

4. Kalman Filter Fusing LSTM for Nonlinear Target Tracking

It can be seen from the above analysis that Kalman filter algorithm requires the linear equation of motion state, and the motion process noise and measurement noise are both Gaussian white noise with 0 mean. However, for nonlinear maneuvering targets, their motion state is often irregular, and the distribution of motion process noise and measurement noise is also a priori unknown. To address these problems, we present the Kalman filter fusing LSTM (KFFLSTM), a novel structure for the nonlinear target tracking. Specifically, we make use of three different LSTM networks to model the nonlinear process function, process noise, and measurement noise in Kalman filter, respectively. Through the deep learning process from massive training data, we are able to achieve the best coefficient representation of the three networks. Then, it is applied to the Kalman filter calculation to obtain a more accurate target state prediction.

4.1. System Description. In essence, our network model is a kind of Kalman filter technology based on deep learning. With the continuous refinement of the measured values, it can regularize the nonlinear target motion state without manually setting the process and measurement models.

Theoretically, by setting the receiving vector sequence of the LSTM network, from Equations (11)–(16), the hidden state of the t-th time step of the LSTM is summarized as

Theoretically, by setting the receiving vector sequence $X = (x_1, x_2, \cdots, x_T)$ of the LSTM network, from Equations (11)–(16), the hidden state of the t-th time step of the LSTM is summarized as

$$h_t = f_{\theta_h}(x_t, h_{t-1}), \tag{17}$$

where h_{t-1} is the hidden state of the previous step and f_{θ_h} is a deterministic gated activation function through the parameter set θ_h determined. The corresponding output at time t can also be summarized as

$$o_t = f_{\theta_o}(x_t, h_{t-1}), \tag{18}$$

where f_{θ_o} is a deterministic gated activation function through the parameter set θ_o determined.

Through the training process of LSTM network, we achieve the optimal parameter set θ^* (θ_h and θ_o are its subsets) in statistical sense, with the error function minimized. With its help, the LSTM deterministically approximate the conditional probability density $p(y_1, \cdots, y_t | x_1, \cdots, x_t)$ and complete the deterministic mapping from input to output. This is expressed as

$$p(y_1, \cdots, y_T | x_1, \cdots, x_T) = \prod_{t=1}^{T} p(y_t | x_t), \tag{19}$$

where each of $p(y_t|x_t)$ is parameterized by a cyclic module, which defines a deterministic function.

Through LSTMs, we can learn the mapping relationship from input to output from abundant training data and apply them to the estimation of key parameters in Kalman filter. Compared with the traditional model-based methods, this new one's advantages will be very significant. In traditional methods, these parameters are often determined by fixed models or obtained by experience, which limits their application in nonlinear and non-Gaussian target tracking. The fusion of LSTM networks and Kalman filter not only makes the Kalman filter algorithm break through the original limitation of linear Gaussian but also helps the updating process of Kalman filter to combine the estimation obtained by LSTM and measurement, which is expected to make the nonlinear target tracking more effective than that of Kalman or LSTM alone.

Specifically, our network model divides the tracking task into three different LSTM networks according to their function. Each LSTM estimates the undetermined matrix in the Kalman filter: the LSTM-A estimates the process state transition matrix A (Equation (6)), the LSTM-Q estimates the process noise covariance matrix Q (Equation (7)), and the LSTM-R estimates the measurement noise covariance matrix R (Equation (8)). The three networks are organically combined by the prediction and update process of the Kalman filter. Compared with one whole LSTM [23] learning process, this arrangement is able to make the learning task of each LSTM simpler, converge faster, and improve the effect of overall estimation.

The system structure is shown in Figure 1. In the time step t, LSTM-A takes the prediction of the previous step $\hat{x}_{t-1}$ as the input to generate the intermediate state of process estimation $\hat{x}'_t$. Then, it is input to LSMT-Q to generate the estimation of process noise covariance $\widehat{Q}_t$. Meanwhile, the observed value y_t of this time step is input to LSTM-R to generate the estimation of the measurement noise covariance $\widehat{R}_t$. Finally, the $\hat{x}'_t$, $\hat{y}_t$, $\widehat{Q}_t$, and $\widehat{R}_t$ are inputted to the above update Equations (8)–(10) of Kalman filter to obtain the final state prediction value $\hat{x}_t$ of this time step. Each time step in the whole prediction process is iterated according to this step, and finally a whole tracking prediction results can be obtained.

4.2. Model Prediction and Update Steps. Assuming that the measured value is a noisy estimation of the basic state, it can be simplified to make $H = I$ in above Equation (5). The original model updates as

$$\begin{aligned} x_t &= Ax_{t-1} + w \quad w \sim N\left(0, \widehat{Q}_t\right), \\ \hat{y}_t &= x_t + v \qquad\quad v \sim N\left(0, \widehat{R}_t\right), \end{aligned} \tag{20}$$

where the process equation A in the prediction step is modeled by LSTM-A network, $\widehat{Q}_t$ is output by LSMT-Q and $\widehat{R}_t$ is output by LSTM-R. $\hat{y}_t$ is the measurement observed at time t as the input to the filter. The formula of

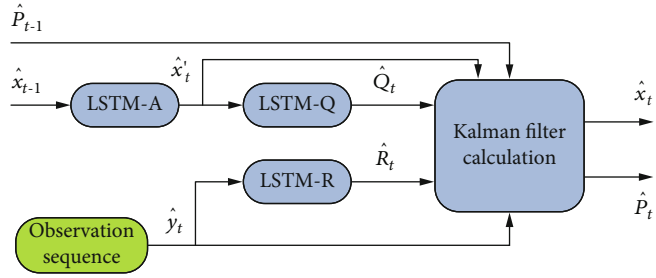

Figure 1: Overview of the KFFLSTM. In a time step t, the architecture of the KFFLSTM is composed of three LSTM modules and Kalman filter calculation modules.

Kalman filter process is modified to

$$\widehat{x}'_t = A\widehat{x}_{t-1}, \tag{21}$$

$$\widehat{P}'_t = A\widehat{P}_{t-1}A^T + \widehat{Q}_t, \tag{22}$$

$$K_t = \widehat{P}'_t\left(\widehat{P}'_t + \widehat{R}_t\right)^{-1}, \tag{23}$$

$$\widehat{x}_t = \widehat{x}'_t + K_t\left(\widehat{y}_t - \widehat{x}'_t\right), \tag{24}$$

$$\widehat{P}_t = (I - K_t)\widehat{P}'_t, \tag{25}$$

where $\widehat{x}_t$ and $\widehat{x}_{t-1}$ represent the posteriori state estimation values at time t and time $t-1$, respectively, which are part of the filtering results. Other filtering results are $\widehat{P}_t$ and $\widehat{P}_{t-1}$, which represent the a posteriori estimation covariance at time t and time $t-1$, respectively. Next section, we give a detailed description of the LSTM networks.

4.3. LSTM Network Architecture. We use LSTM-A, LSTM-Q, and LSTM-R to represent the three LSTM networks. Each network is described in Figure 2.

Specifically, the estimation task of nonlinear process function is the most important, in which a large of features need to be extracted from the training data. Therefore, the LSTM-A is designed consisting of four stacked LSTM layers; each layer has 1024 hidden units. There are three fully connected (FC) layers, whose hidden units are set 1024, 1024, and 2, respectively. The process noise estimation task is relatively easy. The LSTM-Q is designed as a single layer LSTM with 256 hidden units, adding a full connection layer of 2 hidden units. The measurement noise is relatively fixed, and the estimation is easier. The LSTM-R is designed as a single-layer LSTM network with 128 hidden units and a full connection layer connecting 2 hidden units. Behind each LSTM layer in the design, there is a loss layer with a retention probability of 0.8. The activation function of each fully connected layer is the nonlinear function Relu (except the last layer).

It should be noted that in order to ensure the invertibility of the matrix generated when calculating the Kalman gain, the $\widehat{Q}_t$ and $\widehat{R}_t$ are limited to diagonal matrices and positive definite by exponentiation of the outputs of LSTM-Q and LSTM-R.

4.4. Loss Function. In practice, we first used the commonly used sum of squares of residuals (SSR) loss function:

$$L(\theta) = \frac{1}{T}\sum_{t=1}^{T}\|x_t - \widehat{x}_t(\theta)\|^2. \tag{26}$$

However, the value of the loss function did not converge during the training process. After the problem locating, we found that the LSTM-A module cannot learn any reasonable mapping. Therefore, a regular term is added to the loss function to strengthen the gradient flow to LSTM-A. So the modified loss function is as follows:

$$L(\theta) = \frac{1}{T}\sum_{t=1}^{T}\|x_t - \widehat{x}_t(\theta)\|^2 + \lambda\left\|x_t - \widehat{x}'_t(\theta)\right\|^2, \tag{27}$$

where x_t the real target motion state at time t, $\widehat{x}_t(\theta)$ is the result of state estimation values at time t, and $\widehat{x}'_t(\theta)$ is the intermediate result estimated by LSTM-A. The super parameter λ is empirically set to 0.7.

4.5. Training Optimization. Our goal is to find the optimal fixed parameter sets, which are able to make the loss caused by all free parameters in the complex loss function minimized. These parameters are the sum of all weight and bias parameters of all three LSTM modules. The optimization process is often nonconvex. Facing many local minima, we only need to achieve a small enough local minimum. That is adequate for our practice [24].

To train the model, we use the back propagation time algorithm [25] to obtain the gradient. The gradient update is performed according to the commonly used Adam [26] optimizer.

The huge training data set may increase the computational complexity of gradient calculation. In order to overcome this problem, we only use a small batch of training data to update the parameters in each iteration. The benefits are selecting the descent direction accurately for every gradient descent and reducing the training shock, within a certain range. However, the defects are increasing the number of iterations, rising the operation time and slowing parameters modification. There is no final conclusion on how to select the batch size. Too small batch will introduce more noise, and too large batch will increase the training shock [27, 28].

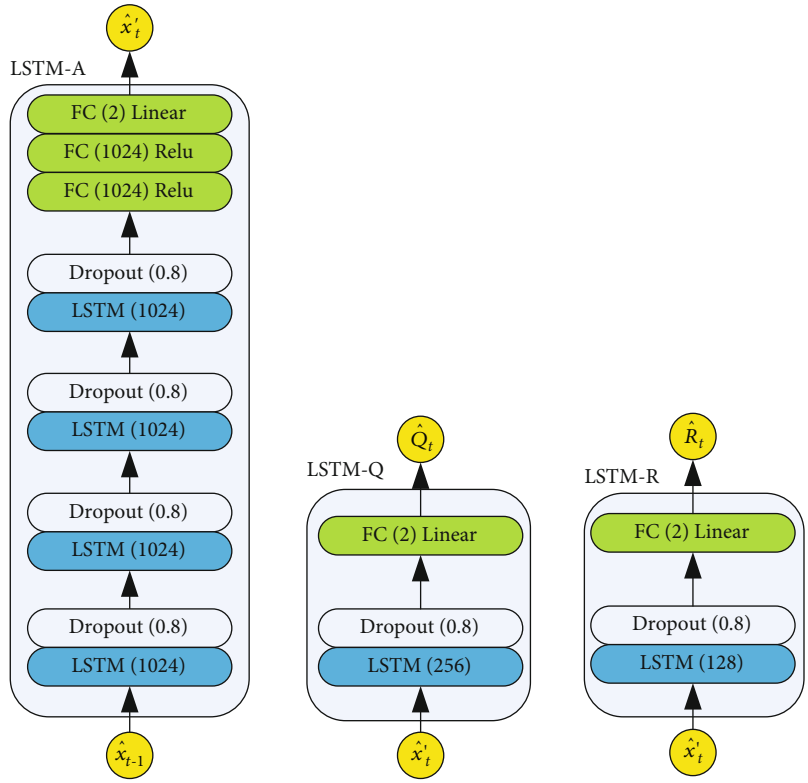

Figure 2: LSTM-A, LSTM-Q, and LSTM-R architecture. As detailed below, the LSTM-A, LSTM-Q, and LSTM-R are composed of different neuron layers in KFFLSTM.

For the sake of suppressing the problem of gradient explosion, the gradient clipping mechanism is introduced. When the weight is updated too quickly in the iteration, it is easy to lead to loss divergence. Therefore, we need to cut the gradient. By controlling the maximum normal form of the gradient, we limit the weight to update in an appropriate range, which effectively restrain the gradient problem.

5. Experiments

In this section, in order to show the tracking performance of the designed KFFLSTM structure for nonlinear targets, we compare the performance with a series of common nonlinear target tracking algorithms, including EKF, UKF, PF, IMM, and B-LSTM structure. This B-LSTM structure, proposed in [19], is the representative of a kind of model, whose feature is using two whole LSTMs to learn the prediction step and measurement update step from the training data, respectively.

5.1. Dataset. Specifically, we use a typical nonlinear time series model. This model is widely used in the literature as a benchmark numerical filtering technique to demonstrate performance improvement [29, 30]. The state space equation is as follows:

$$x_k = \frac{x_{k-1}}{2} + 25\frac{x_{k-1}}{1+x_{k-1}^2} + 8\cos(1.2k) + u_k,$$
$$z_k = \frac{x_k^2}{2} + v_k, \tag{28}$$

where the process noise distribution is $u_k \sim N(0, \sigma_u^2)$, $\sigma_u^2 = 10$. The observed noise distribution is $v_k \sim N(0, \sigma_v^2)$, $\sigma_v^2 = 1$. The initial state distribution is $x_0 \sim N(0, 10)$.

Through the simulation of the model, the training data set can be obtained. Specifically, the random observation target was generated within a certain observation time, and the initial position was randomly set within a certain range. We produced 64 random path observation data, respectively, by simulating the target state sequence and the corresponding observation sequence, resulting in a total of 65536×2 observation data. It was assumed that the data variable of interest was set as the position state of the observation target.

In the implementation of LSTM network structure, adopting the network structure mentioned above, the scale of network parameters is nearly 307460 parameters, whose best values need to be achieved by training process using the loss function and optimization method described in Section 3. The initial learning rate of training KFFLSTM structures is set to $1e-5$ and attenuates to 0.9 from the second cycle. For this training, we use reduced time back propagation to propagate gradients of 64 time steps, and the batch size is set to 16. In the test stage, we simulated the test data set containing 1024 target state sequences and corresponding observation sequences to verify the enhancement of performance.

5.2. Performance Comparison. In the test, we selected some traditional methods, widely used to solve the nonlinear target tracking problem [29], EKF, UKF, IMM, PF, and the single LSTM method B-LSTM as the baseline to compare

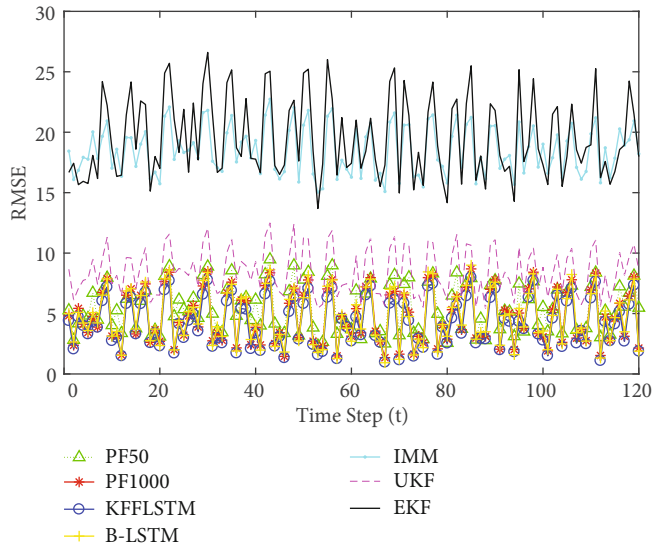

Figure 3: The RMSE of KFFLSTM compared with other baseline methods. Our method performs better than the traditional method and single LSTM method.

with the designed KFFLSTM structure. The IMM method selected CV and CT models to switch. The PF method simulated the performance of 50 particles and 1000 particles, respectively. These above algorithms all have performed 500 Monte Carlo calculations and obtained the root mean square error (RMSEs) of each time step, as shown in Figure 3.

As shown in Figure 3, for this typical nonlinear time series target tracking, EKF performs worse than sampling methods (UKF and PF) and LSTM-based methods. The multimodel method IMM only selects the commonly used CV and CT models for switching, so its performance is similar to EKF. For the PF method, the performance of 1000 particles is slightly worse than that of B-LSTM method. Our proposed KFFLSTM performs better than B-LSTM. It is worth noting that although the traditional methods are very different from the proposed LSTM network methods in solving the nonlinear target tracking problem, the fluctuation of RMSE curve is similar. This shows that these methods can achieve the purpose of tracking, but the accuracy is different.

In addition, we also compared the average running time and average RMSE of each iteration of various methods under the condition of using the same CPU, as shown in Table 1.

Because the estimation accuracy of EKF, UKF, and IMM is far from that of other methods, we focus on the comparison between PF and LSTM methods. It can be seen that our proposed KFFLSTM can provide accurate estimation in a short calculation time. Compared with PF with 50 particles, the accuracy is improved by 27.3% and the calculation time is shortened by 45.5%. Compared with 1000 particle PF, the accuracy is improved by 14.5%, and the calculation time is reduced by 96.8%. Compared with the B-LSTM method, this method improves the accuracy by 13.2%, but the running time is slightly longer. This may be because KFFLSTM not only needs to calculate a similar number of gradients but also has to calculate many matrix operations, including the inversion of the square matrix in Equation (23), at each time step.

Table 1: Table of the main algorithms, RMSE values, and runtimes at each iteration.

Algorithms	RMSE	Run times
PF (50 particles)	5.532	1.424 ms
PF (1000 particles)	4.758	24.32 ms
KFFLSTM	4.235	0.775 ms
B-LSTM	4.732	0.453 ms

5.3. Influence of Different Noise Variances. In order to carry out this experiment, we change the observation noise variance and motion noise variance, respectively, to dynamically simulate the impact of mobility change or measurement error change on the estimation effect of these methods.

Firstly, our simulation uses the same initial state and fixed motion noise variance to calculate the average RMSE of PF50, PF1000, B-LSTM, and KFFLSTM methods, when the observed noise distribution variance is set to the typical value of 1, 10, 50, and 100, as shown in Figure 4.

As shown in Figure 4, when the observation variance increases by order of magnitude, the RMSE of PF1000 changes a little and increases slightly, while the RMSE of PF50 changes greatly and increases significantly. The RMSE of B-LSTM and KFFLSTM nearly remain the same.

Secondly, our simulation uses the same initial state and fixed the observation noise distribution variance to calculate the average RMSE of PF50, PF1000, B-LSTM, and KFFLSTM methods, when the motion noise variance is set to the typical value of 1, 10, 50, and 100, as shown in Figure 5.

As shown in Figure 5, when the motion variance increases by order of magnitude, the RMSE change of PF1000 and PF50 increases significantly. The RMSE of B-LSTM increases slowly, and KFFLSTM increases slightly.

This shows that the two LSTM-based methods are insensitive to the mobility change of nonlinear tracking target and the change of measurement error compared with the

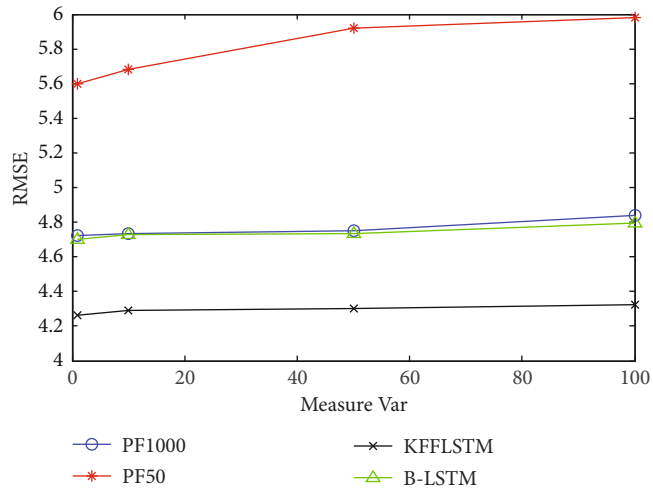

Figure 4: The RMSE of KFFLSTM with different measure noise variance. RMSE curve of our method and B-LSTM increases slightly.

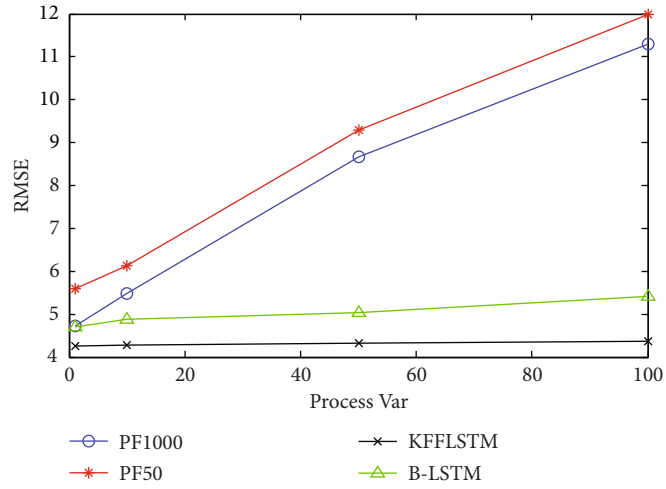

Figure 5: The RMSE of KFFLSTM with different process noise variance. Our method's RMSE curve basically remained unchanged.

traditional methods. Especially, the antisensitivity of KFFLSTM described in this paper is stronger than that of LSTM alone method.

5.4. Influence of Initial a Prior. In our experiment, we assumed that the initial a priori motion state of the target is known, that was to say, the initial a priori of the test data and the training data were considered to be the same. However, in practice, the initial state of the actual target is often unexpected. Once it is out of the distribution range of training data, this will lead to poor tracking effect of the model on test data [30]. In order to solve this problem, we may increase the support range of the initial a priori of training data theoretically, but the price is more training data required.

To verify this, we simulated the training data with different initial state distribution range and the same test data set and calculated the RMSE of the KFFLSTM, respectively, as shown in Table 2. The results show that the estimation accuracy of the model remains almost unchanged as long as the initial prior distribution of the training data can cover the initial prior distribution of the test data.

Table 2: RMSE of the KFFLSTM trained with different initial prior data sets.

Training data initial prior distribution	Test data initial prior distribution	RMSE of KFFLSTM
N (0, 10)	N (0, 10)	4.235
N (0, 100)	N (0, 10)	4.238
N (0, 500)	N (0, 10)	4.237
N (0, 1000)	N (0, 10)	4.239

6. Conclusion

Against nonlinear moving target tracking, KFFLSTM structure combining LSTM and Kalman filter is proposed in this

paper. This creative structure is able to learn the nonlinear motion process, motion noise, and measurement noise from massive training data and then apply it to the Kalman filter equations to complete the accurate estimation of the target motion states. It not only does not need to specify the motion model and noise model in advance but also is capable of learning the nonlinear motion model through the data, which is very difficult to record explicitly. The experimental data indicate that compared to the traditional methods, this method estimates the states more accurately within a shorter calculation time and also performs better than the single LSTM method.

What this paper provides is only a preliminary algorithm. In the future, with the continuous development of LSTM or RNN, its performance in the field of nonlinear tracking will be continuously improved.

Data Availability

The experimental data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This research was supported by the Fundamental Research Funds for the Central Universities under Grant 3102019ZX015, the Aeronautical Science Foundation of China under Grant 2019ZH0T7001, the Scientific Research Foundation of Xi'an Aeronautical University under Grant 2019KY0207, and the Fundamental Research Funds for the Central Universities under Grant D5000220131.

References

[1] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[2] H. Kushner, "Approximations to optimal nonlinear filters," *IEEE Transactions on Automatic Control*, vol. 12, no. 5, pp. 546–556, 1967.

[3] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "A new approach for filtering nonlinear systems," in *Proceedings of 1995 American Control Conference-ACC'95. IEEE*, pp. 1628–1632, Seattle, WA, USA, 1995.

[4] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[5] X. R. Li, X. Zhi, and Y. Zhang, "Multiple-model estimation with variable structure. V. likely-model set algorithm," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 36, no. 2, pp. 448–466, 2000.

[6] X. Wang, S. Challa, R. J. Evans, and X. R. Li, "Minimal submodel-set algorithm for maneuvering target tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 4, pp. 1218–1231, 2003.

[7] P. Ondruska and I. Posner, "Deep tracking: seeing beyond seeing using recurrent neural networks," in *Proceedings of the Thirtieth AAAI Conference on Artifificial Intelligence*, pp. 3361–3367, Phoenix, Arizona, 2016.

[8] M. Q. Liu, Z. L. Liu, W. D. Lu, Y. Chen, X. Gao, and N. Zhao, "Distributed few-shot learning for intelligent recognition of communication jamming," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 3, pp. 395–405, 2022.

[9] M. Q. Liu, B. Li, Y. F. Chen et al., "Location parameter estimation of moving aerial target in space-air-ground integrated networks-based IoV," *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 5696–5707, 2022.

[10] M. Q. Liu, C. H. Liu, M. Li, Y. Chen, S. Zheng, and N. Zhao, "Intelligent passive detection of aerial target in space-air-ground integrated networks," *China Communications*, vol. 19, no. 1, pp. 52–63, 2022.

[11] M. Q. Liu, J. K. Wang, N. Zhao, Y. Chen, H. Song, and R. Yu, "Radio frequency fingerprint collaborative intelligent identification using incremental learning," *IEEE Transactions on Network Science and Engineering*, 2021.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[13] C. Gao, H. Liu, S. Zhou et al., "Maneuvering target tracking with recurrent neural networks for radar application," in *2018 International Conference on Radar (RADAR)*, pp. 1–5, Brisbane, QLD, Australia, 2018.

[14] T. Haarnoja, A. Ajay, S. Levine, and P. Abbeel, "Backprop KF: learning discriminative deterministic state estimators," *Advances in Neural Information Processing Systems*, vol. 29, pp. 1132–1143, 2016.

[15] H. Coskun, F. Achilles, R. DiPietro, N. Navab, and F. Tombari, "Long short-term memory Kalman filters: recurrent neural estimators for pose regularization," in *Proceedings of the 2017 IEEE International Conference on Computer Vision*, pp. 5525–5533, Venice, Italy, 2017.

[16] C. Gao, J. Yan, S. Zhou, B. Chen, and H. Liu, "Long short-term memory-based recurrent neural networks for nonlinear target tracking," *Signal Processing*, vol. 164, pp. 67–73, 2019.

[17] J. P. Llerena Caña, J. García Herrero, and J. M. Molina López, "Forecasting nonlinear systems with LSTM: analysis and comparison with EKF," *Sensors*, vol. 21, no. 5, pp. 1805–1820, 2021.

[18] Z. Zhang, M. Hou, F. Zhang, and C. R. Edwards, "An LSTM based Kalman filter for spatio-temporal ocean currents assimilation," in *Proceedings of the International Conference on Underwater Networks & Systems*, pp. 1–7, Atlanta, GA, USA, 2019.

[19] F. Song, Y. Li, Y. Bi, and M. Li, "Radar maneuvering target tracking based on LSTM network," in *Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery*, pp. 1780–1791, Springer, Cham, 2021.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[22] K. Greff, R. K. Srivastava, J. Koutník, and R. Bas, "LSTM: a search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 9, no. 7, pp. 1031–1047, 2016.

[23] R. G. Krishnan, U. Shalit, and D. Sontag, "Deep kalman filters," 2015, https://arxiv.org/abs/1511.05121.

[24] A. Tarsauliya, S. Kant, R. Kala, R. Tiwari, and A. Shukla, "Analysis of artificial neural network for financial time series forecasting," *International Journal of Computer Applications*, vol. 9, no. 5, pp. 16–22, 2010.

[25] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[26] D. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2014, https://arxiv.org/abs/1412.6980.

[27] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: generalization gap and sharp minima," 2017, https://arxiv.org/abs/1609.04836.

[28] S. L. Smith, P. J. Kindermans, C. Ying, and Q. V. Le, "Don't decay the learning rate, in crease the batch size," *Proceedings of the International Conference on Learning Representations*, vol. 1, pp. 1280–1293, 2018.

[29] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[30] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine Learning*, vol. 79, no. 1-2, pp. 151–175, 2010.