Received January 19, 2020, accepted February 10, 2020, date of publication February 14, 2020, date of current version March 3, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2974039

The Research on Distributed Fusion Estimation Based on Machine Learning

ZHENGXIAO PENG[1,2], YUN LI[3], AND GANG HAO[1,2]
[1]School of Electronic Engineering, Heilongjiang University, Harbin 150080, China
[2]Key Laboratory of Information Fusion Estimation and Detection, Heilongjiang University, Harbin 150080, China
[3]School of Computer and Information Engineering, Harbin University of Commerce, Harbin 150001, China

Corresponding author: Gang Hao (haogang@hlju.edu.cn)

This work was supported in part by the National Natural Science Foundation (NNSF) of China under Grant 61503127 and Grant 51477109, in part by the University Basic Research foundation of Heilongjiang Province under Grant KJCX201901, in part by the Outstanding Youth Foundation of Heilongjiang University, in part by the Key Laboratory of Information Fusion Estimation and Detection, Heilongjiang Province, in part by the Harbin University of Commerce Ph.D. Research Startup Project under Grant 2019DS030, and in part by the Science Foundation of Harbin Commerce University under Grant 18XN064 and Grant 17XN060.

ABSTRACT Multi-sensor distributed fusion estimation algorithms based on machine learning are proposed in this paper. Firstly, using local estimations as inputs and estimations of three classic distributed fusion (weighted by matrices, by diagonal matrices and by scalars) as the training sets, three distributed fusion algorithms based on BP network (BP net-based fusion weighted by matrices, by diagonal matrices and by scalar) are proposed and the selection basis of the number of nodes in hidden layer is given. Furthermore, by using local estimations as inputs and centralized fusion estimation as training set, another recurrent net-based distributed fusion algorithm is proposed, in the case that neither true states nor cross-covariance matrices is available. This method is not limited to the linear minimum variance (LMV) criterion, so its accuracy is higher than the classical three distributed fusion algorithms. A radar tracking simulation verifies the effectiveness of the proposed fusion networks.

INDEX TERMS Distributed fusion, machine learning, recurrent networks, BP network.

I. INTRODUCTION

With the development of sensor technology, the performance of sensors has been continuously improved, and the number of sensors integrated in the systems has been continuously increased. To some extent, the number of sensors even represents the accuracy and reliability of the system. With the increase in the number of sensors, information fusion has become an indispensable part of information processing for multi-sensor systems and has attracted many attentions of scholars.

Multi-sensor fusion estimation [1] is an important branch of information fusion and it often works at the bottom of the fusion framework. For fusion estimation based on Kalman filter, there are two basic fusion frameworks: centralized fusion and distributed fusion [2]. Centralized fusion needs to transmit measurement data to fusion center and then estimate using the expanded measurement. It can give the globally optimal estimation, but the large communicational and computational cost and the poor fault tolerance are its disadvantages. Distributed fusion can estimate in local node according to the local measurements and then transmit the local estimates to fusion center. It can only give the suboptimal estimation [3], [4], but due to the lower communicational and computational cost, it is more suitable for large-scale sensor network systems. Distributed fusion is widely used in many system [5], [6], such as the famous Carlson federal Kalman filter [7], [8].

Based on the LMV criterion, a optimal fusion algorithm weighted by matrices using weighted least squares method is proposed (i.e., Carlson's federated Kalman filter) [7], [8]. Roy and Iltis proposed a decentralized static filter for the linear system with correlated measurement noises [9]. Kim proposed the multi-sensor optimal information fusion estimate in the maximum likelihood sense under the assumption of normal distribution [10]. Based on the LMV criterion, fusion algorithm weighted by matrices, scalar and diagonal matrices was proposed by Sun and Deng [11]–[13]. All these distributed fusion algorithms have lower communicational and computational cost and good fault tolerance, and are widely used in various fields.

The associate editor coordinating the review of this manuscript and approving it for publication was Ludovico Minati.

Artificial Neural Network (ANN) is an algorithmic mathematical model for distributed parallel information processing which imitates the characteristics of information transmission and reflective behavior of human brain nervous system [14]. Due to the good approximation performance of ANN, it has been well applied in the state estimation and information fusion [15]–[21]. Rao et al proposed various learning-based estimators to solve the fusion estimation problem (i.e., Artificial Neural Networks (ANNs), the Nadaraya-Watson estimator and the Nearest Neighbor Projective Fuser) [22]–[24]. Chowdhury proposes a neural data fusion method, which is performed by a set of artificial neurons and synaptic weights are used as weight estimates for optimal data fusion [20]. Liu *et al.* [25] and Brigham *et al.* [26] proposed the learning-based nonlinear fusion algorithms, and Brigham et al studied and compared the use of the following cages: ANNs, support vector regression (SVR), Nadaraya-Watson (NW) estimator and nearest neighbor (NN) projection fuser [26].

BP (Back Propagation) algorithm is first proposed by Werbos in his doctoral thesis, which provides a practical solution for the training and implementation of multi-layer neural network [27]–[29]. Rumelhart et al dissect the error back propagation algorithm of the multi-layer network, which further promotes the BP algorithm [30]. The topological structure of BP network includes input layer, hidden layer and output layer which can store a certain mapping relationship by learning without knowing the specific mathematical expression of input and output beforehand. Cybenko et al successively analyze the approximation performance of BP network, and prove that the continuous feedforward neural networks with single hidden layer and sigmoid transfer function can approximate any complex continuous mapping with arbitrary precision [31]–[33]. For recurrent networks (Hopfield, Elman, CG, BSB, CHNN, DHNN, etc.), the output are not only related to the current input, but also related to the previous input of the network, which makes them have an iterative structure similar to the Kalman filtering framework [34], [35], [40].

The main work of this paper is to construct multi-sensor distributed fusion frameworks based on ANN and machine learning. Firstly, using local estimations as the inputs and estimations of three classic distributed fusion (weighted by matrices, by diagonal matrices and by scalars) as the training sets, three distributed fusion algorithms based on BP network (BP net-based fusion weighted by matrices, by diagonal matrices and by scalar) are proposed and the selection basis of the number of nodes in hidden layer is given. Furthermore, the distributed fusion algorithm based on BP network (BP net-based distributed fusion algorithm) is analyzed, which uses local estimations as network inputs and true states as the training set, in the case that the true states are available but cross-covariance matrices are not. Finally, using local estimations as inputs and centralized fusion estimation as training set, another recurrent net-based distributed fusion algorithm is proposed, in the case that neither true states nor cross-covariance matrices is available. This method is not limited to the LMV criterion, so its accuracy is higher than the classical three distributed fusion algorithms.

Notations: $\Re^n$ denotes the n-dimensional Euclidean space; $\boldsymbol{I}_{n\times n}$ is the n-dimensional identity matrix; 'E' denotes the mathematical expectation; Superscripts 'T' and '-1' denote the transpose and inverse, respectively; δ_{tk} is the Kronecker delta function (i.e., $\delta_{tt} = 1$ and $\delta_{tk} = 0(t \neq k)$); $\mathrm{tr}\boldsymbol{P}$ represents the trace of the matrix $\boldsymbol{P}$; $\hat{\boldsymbol{x}}^{(j)}_{k|k-\gamma} = \mathrm{E}\left\{\boldsymbol{x}_k | z^{(j)}_{0\sim k-\gamma}\right\}$ $(\gamma = 0, \cdots, k)$ is the estimator of $\boldsymbol{x}_k$ based on the measurements $z^{(j)}_{0\sim k-\gamma} = \left\{z^{(j)}_0, \cdots, z^{(j)}_{k-\gamma}\right\}$; $\tilde{\boldsymbol{x}}^{(*)}_{k|k-\gamma} = \boldsymbol{x}_k - \hat{\boldsymbol{x}}^{(*)}_{k|k-\gamma}$ $(* = j, M, S, D, C)$ is the estimation error; $\boldsymbol{P}^{(ij)}_{k|k-\gamma} = \mathrm{E}\left\{\left(\tilde{\boldsymbol{x}}^{(i)}_{k|k-\gamma}\middle| z^{(i)}_{0\sim k-\gamma}\right)\left(\tilde{\boldsymbol{x}}^{(j)}_{k|k-\gamma}\middle| z^{(j)}_{0\sim k-\gamma}\right)^{\mathrm{T}}\right\}$ is the estimation error cross-covariance matrix and $\boldsymbol{P}^{(ii)}_{k|k-\gamma}$ will be abbreviated to $\boldsymbol{P}^{(i)}_{k|k-\gamma}$.

II. PROBLEM FORMULATION

Consider the discrete linear time-invariant system with multiple sensors

$$\boldsymbol{x}_{k+1} = \boldsymbol{\Phi}\boldsymbol{x}_k + \boldsymbol{\Gamma}\boldsymbol{w}_k \tag{1}$$

$$z^{(j)}_k = \boldsymbol{H}^{(j)}\boldsymbol{x}_k + \boldsymbol{v}^{(j)}_k, \quad j = 1, \cdots, L \tag{2}$$

where $\boldsymbol{x}_k \in \Re^n$ is the state; $z^{(j)}_k \in \Re^{m_j}$ $(j = 1, \cdots, L)$ are the measurements; $\boldsymbol{w}_k \in \Re^r$ is the process noise; $\boldsymbol{v}^{(j)}_k \in \Re^{m_j}$ $(j = 1, \cdots, L)$ are the measurements noises; $\boldsymbol{\Phi}$, $\boldsymbol{\Gamma}$, $\boldsymbol{H}^{(j)}$ are time-invariant matrices with suitable dimensions.

Assumption $\boldsymbol{w}_k$, $\boldsymbol{v}^{(j)}_k$ $(j = 1, \cdots, L)$ are the uncorrelated white noise with zero mean and covariance

$$\mathrm{E}\left\{\begin{bmatrix}\boldsymbol{w}_k \\ \boldsymbol{v}^{(i)}_k\end{bmatrix}\begin{bmatrix}\boldsymbol{w}^{\mathrm{T}}_t & \left(\boldsymbol{v}^{(j)}_t\right)^{\mathrm{T}}\end{bmatrix}\right\} = \begin{bmatrix}\boldsymbol{Q} & 0 \\ 0 & \boldsymbol{R}^{(i)}\delta_{ij}\end{bmatrix}\delta_{kt}, i, j = 1, \cdots, L \tag{3}$$

Our aim is to obtain the estimation $\hat{\boldsymbol{x}}_{k|k}$ as close to the $\boldsymbol{x}_k$ as possible, based on the measurements $\left(z^{(j)}_0 \cdots z^{(j)}_k\right)$ $(j = 1, \cdots, L)$. There are two basic structures for this type of fusion estimation, one is centralized fusion estimation framework and the other is distributed fusion estimation framework. When we use neural networks for fusion and estimation, the two fusion estimation frameworks are shown in FIGURE 1 and FIGURE 2, respectively. In the paper, we choose the distributed fusion estimation framework for the following two reasons:

1) The centralized fusion estimation framework shown in FIGURE 1 needs to transmit the original measurements. Due to the limitations of transmission bandwidth and energy, the sensors outputs are required to be simple, which makes many sensors unavailable, such as imaging sensors. However, the communication network in the distributed fusion estimation framework only needs to transmit the interested states, which makes the networks more efficient and fast.

2) The Kalman filter is optimal for systems (1) and (2) [38]. It means that, for distributed fusion estimation framework

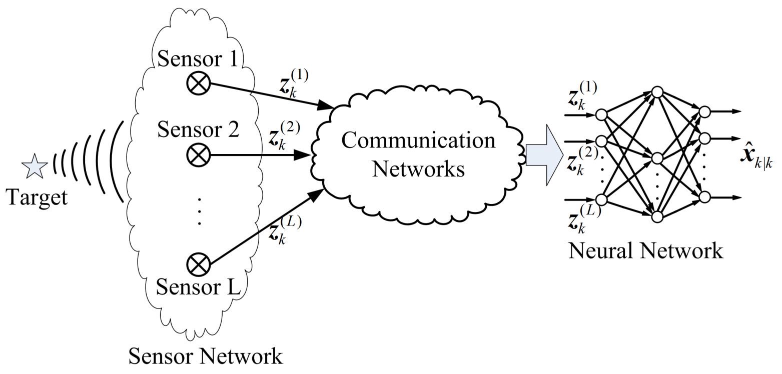


FIGURE 1. Centralized fusion estimation framework.

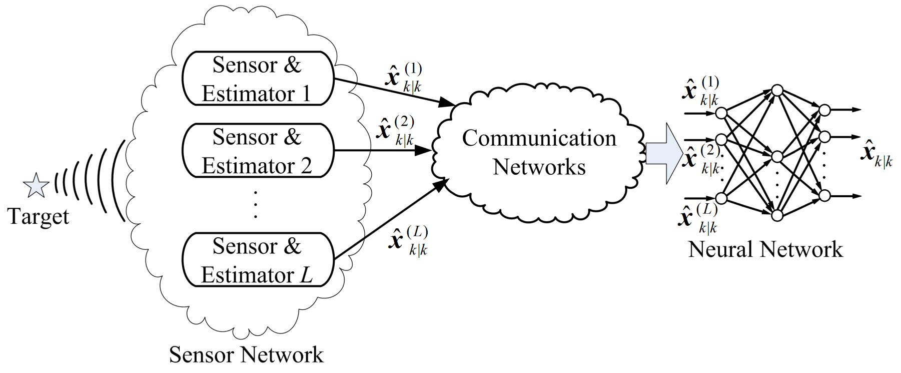


FIGURE 2. Distributed fusion estimation framework.

shown in FIGURE 2, there is no loss of information before neural network and neural network only need to fuse local estimates. However, neural network in the centralized fusion estimation framework need to process estimation and fusion simultaneously, which needs the neural network to have an iterative framework (i.e., a neural network with feedback [39]) and more nodes in hidden layer. The complexity of the neural network will bring troubles to construction and training, and more computational cost to the operation of the neural network.

III. CLASSICAL FUSION ALGORITHMS

Lemma 1 [11]: (The optimal fusion algorithm weighted by matrices in the sense of LMV) Let $\hat{\boldsymbol{x}}_{k|k}^{(j)}$ $(j = 1, \cdots, L)$ be unbiased estimators of $\boldsymbol{x}_k \in \Re^n$ based on the measurements $\left(z_0^{(j)} \cdots z_k^{(j)}\right)$. Then the optimal fusion estimator $\hat{\boldsymbol{x}}_{k|k}^{(M)}$ weighted by matrices in the sense of LMV is given by:

$$\hat{\boldsymbol{x}}_{k|k}^{(M)} = \sum_{j=1}^{L} \boldsymbol{A}_k^{(j)} \hat{\boldsymbol{x}}_{k|k}^{(j)} \tag{4}$$

where the optimal matrix weights $\boldsymbol{A}_k^{(j)}$ are computed by

$$\left[\boldsymbol{A}_k^{(1)}, \cdots, \boldsymbol{A}_k^{(L)}\right] = \left(\boldsymbol{e}^{\mathrm{T}} \boldsymbol{P}_{k|k}^{-1} \boldsymbol{e}\right)^{-1} \boldsymbol{e}^{\mathrm{T}} \boldsymbol{P}_{k|k}^{-1} \tag{5}$$

where $\boldsymbol{P}_{k|k}$ is an $nL \times nL$ symmetric positive definite matrix, and $\boldsymbol{e}$ is $nL \times n$ matrix:

$$\boldsymbol{P}_{k|k} = \begin{bmatrix} \boldsymbol{P}_{k|k}^{(1)} & \cdots & \boldsymbol{P}_{k|k}^{(1L)} \\ \vdots & \ddots & \vdots \\ \boldsymbol{P}_{k|k}^{(L1)} & \cdots & \boldsymbol{P}_{k|k}^{(L)} \end{bmatrix}, \boldsymbol{e} = \begin{bmatrix} \boldsymbol{I}_n \\ \vdots \\ \boldsymbol{I}_n \end{bmatrix} \tag{6}$$

The corresponding fusion filtering error covariance matrix $\boldsymbol{P}_{k|k}^{(M)}$ is:

$$\boldsymbol{P}_{k|k}^{(M)} = \left(\boldsymbol{e}^{\mathrm{T}} \boldsymbol{P}_{k|k}^{-1} \boldsymbol{e}\right)^{-1} \tag{7}$$

and

$$\operatorname{tr} \boldsymbol{P}_{k|k}^{(M)} \leq \operatorname{tr} \boldsymbol{P}_{k|k}^{(j)}, \quad j = 1, \cdots, L \tag{8}$$

Lemma 2 [12]: (The optimal fusion algorithm weighted by scalars in the sense of LMV) Let $\hat{\boldsymbol{x}}_{k|k}^{(j)}$ $(j = 1, \cdots, L)$ be unbiased estimators of $\boldsymbol{x}_k \in \Re^n$ based on the measurements $\left(z_0^{(j)} \cdots z_k^{(j)}\right)$. Then the optimal fusion estimator $\hat{\boldsymbol{x}}_{k|k}^{(S)}$ weighted by scalars in the sense of LMV is given by:

$$\hat{\boldsymbol{x}}_{k|k}^{(S)} = \sum_{j=1}^{L} a_k^{(j)} \hat{\boldsymbol{x}}_{k|k}^{(j)} \tag{9}$$

where the optimal fusion scalar weights $\boldsymbol{a}_k = [a_k^{(1)} \cdots a_k^{(L)}]$ is calculated by:

$$\boldsymbol{a}_k = \frac{\boldsymbol{e}^{\mathrm{T}}\left(\boldsymbol{P}_{k|k}^{\mathrm{tr}}\right)^{-1}}{\boldsymbol{e}^{\mathrm{T}}\left(\boldsymbol{P}_{k|k}^{\mathrm{tr}}\right)^{-1}\boldsymbol{e}} \tag{10}$$

where $\boldsymbol{P}_{k|k}^{\mathrm{tr}}$ is the $L \times L$ positive definite matrix, and $\boldsymbol{e}$ is $L \times 1$ vectors:

$$\boldsymbol{P}_{k|k}^{\mathrm{tr}} = \begin{bmatrix} \mathrm{tr}\,\boldsymbol{P}_{k|k}^{(1)} & \cdots & \mathrm{tr}\,\boldsymbol{P}_{k|k}^{(1L)} \\ \vdots & \ddots & \vdots \\ \mathrm{tr}\,\boldsymbol{P}_{k|k}^{(L1)} & \cdots & \mathrm{tr}\,\boldsymbol{P}_{k|k}^{(L)} \end{bmatrix}, \quad \boldsymbol{e} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \tag{11}$$

The fusion filtering error covariance matrix $\boldsymbol{P}_{k|k}^{(S)}$ is:

$$\boldsymbol{P}_{k|k}^{(S)} = \sum_{i}^{L}\sum_{j}^{L} a_i a_j \boldsymbol{P}_{k|k}^{(ij)} \tag{12}$$

and

$$\mathrm{tr}\,\boldsymbol{P}_{k|k}^{(S)}(t|t) \le \mathrm{tr}\,\boldsymbol{P}_{k|k}^{(j)}, j = 1, \cdots, L \tag{13}$$

Lemma 3 [13]: (The optimal fusion algorithm weighted by diagonal matrices in the sense of LMV) Let $\hat{\boldsymbol{x}}_{k|k}^{(j)}$ $(j = 1, \cdots, L)$ be unbiased estimators of $\boldsymbol{x}_k \in \Re^n$ based on the measurements $\left(z_0^{(j)} \cdots z_k^{(j)}\right)$. Then the optimal fusion estimator $\hat{\boldsymbol{x}}_{k|k}^{(D)}$ weighted by diagonal matrices in the sense of LMV is given by:

$$\hat{\boldsymbol{x}}_{k|k}^{(D)} = \sum_{j=1}^{L} \boldsymbol{A}_k^{(j)} \hat{\boldsymbol{x}}_{k|k}^{(j)} \tag{14}$$

where the diagonal matrix weights are calculated by:

$$\boldsymbol{A}_k^{(j)} = \begin{bmatrix} a_{1,k|k}^{(j)} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & a_{n,k|k}^{(j)} \end{bmatrix} \tag{15}$$

The $\boldsymbol{x}_k$, $\hat{\boldsymbol{x}}_{k|k}^{(j)}$ and $\hat{\boldsymbol{x}}_{k|k}^{(D)}$ are rewritten as:

$$\boldsymbol{x}_k = \left[x_{1,k}, \ \cdots, \ x_{n,k}\right]^{\mathrm{T}}, \hat{\boldsymbol{x}}_{k|k}^{(j)} = \left[\hat{x}_{1,k|k}^{(j)}, \ \cdots, \ \hat{x}_{n,k|k}^{(j)}\right]^{\mathrm{T}},$$
$$\hat{\boldsymbol{x}}_{k|k}^{(D)} = \left[\hat{x}_{1,k|k}^{(D)}, \ \cdots, \ \hat{x}_{n,k|k}^{(D)}\right]^{\mathrm{T}} \tag{16}$$

then Equation (14) can be rewritten as:

$$\hat{x}_{i,k|k}^{(D)} = \sum_{j=1}^{L} a_{i,k|k}^{(j)} \hat{x}_{i,k|k}^{(j)}, i = 1, \cdots, n, \quad j = 1, \cdots, L \tag{17}$$

where $\boldsymbol{a}_{i,k|k} = \left[a_{i,k|k}^{(1)}, \cdots, a_{i,k|k}^{(L)}\right]$ are calculated by:

$$\boldsymbol{a}_{i,k|k} = \frac{\boldsymbol{e}^{\mathrm{T}}\left(\boldsymbol{P}_{ii,k|k}\right)^{-1}}{\boldsymbol{e}^{\mathrm{T}}\left(\boldsymbol{P}_{ii,k|k}\right)^{-1}\boldsymbol{e}}, \boldsymbol{e} = [1, \cdots, 1]^{\mathrm{T}} \tag{18}$$

where $\boldsymbol{P}_{ii,k|k}$ is the $L \times L$ positive definite matrix:

$$\boldsymbol{P}_{ii,k|k} = \begin{bmatrix} P_{ii,k|k}^{(1)} & \cdots & P_{ii,k|k}^{(1L)} \\ \vdots & \ddots & \vdots \\ P_{ii,k|k}^{(L1)} & \cdots & P_{ii,k|k}^{(L)} \end{bmatrix} \tag{19}$$

where $P_{ii,k|k}^{(kj)}$ are the $i \times i$ element of $\boldsymbol{P}_{k|k}^{(kj)}$. The fusion filtering error covariance $P_{i,k|k}^{(D)} = \mathrm{E}\left\{\left(x_{i,k} - \hat{x}_{i,k|k}^{(D)}\right)\left(x_{i,k} - \hat{x}_{i,k|k}^{(D)}\right)^{\mathrm{T}}\right\}$ is:

$$P_{i,k|k}^{(D)} = \left[\boldsymbol{e}^{\mathrm{T}}\left(\boldsymbol{P}_{ii,k|k}\right)^{-1}\boldsymbol{e}\right]^{-1}, i = 1, \cdots, n \tag{20}$$

and

$$\mathrm{tr}\,\boldsymbol{P}_{k|k}^{(D)} = \sum_{i=1}^{n} P_{i,k|k}^{(D)} \le \mathrm{tr}\,\boldsymbol{P}_{k|k}^{(j)}, \quad j = 1, \cdots, L \tag{21}$$

Lemma 4 [3]: (Centralized fusion) The measurement function of centralized fusion is

$$z_k^{(C)} = \boldsymbol{H}^{(C)}\boldsymbol{x}_k + \boldsymbol{v}_k^{(C)}, j = 1, \cdots, L \tag{22}$$

where

$$z_k^{(C)} = \left[\left(z_k^{(1)}\right)^{\mathrm{T}} \quad \cdots \quad \left(z_k^{(L)}\right)^{\mathrm{T}}\right]^{\mathrm{T}} \tag{23}$$
$$\boldsymbol{H}^{(C)} = [\boldsymbol{H}_1^{\mathrm{T}} \quad \cdots \quad \boldsymbol{H}_L^{\mathrm{T}}]^{\mathrm{T}} \tag{24}$$
$$\mathrm{cov}(\boldsymbol{v}_k^{(C)}) = \mathrm{diag}(\boldsymbol{R}^{(1)} \quad \cdots \quad \boldsymbol{R}^{(L)}) \tag{25}$$

and 'diag' representation diagonal matrix. For the centralized fusion system with Equations (1) and (22), using Kalman filter, the centralized fusion filter $\hat{\boldsymbol{x}}_{k|k}^{(C)}$ can be yield.

IV. FUSION ALGORITHMS BASED ON BP NETWORK

A. BP NET-BASED FUSION ALGORITHMS TRAINED BY CLASSICAL FUSION ESTIMATIONS

The topology of BP network with single hidden layer is shown in FIGURE 3 [36]. It can store the mapping relationship by learning without knowing the specific mathematical expression of input and output beforehand. In 1989, Cybenko et al analysis the nonlinear function approximation performance of BP neural network, and proved that the continuous feedforward neural network with single hidden layer and transfer function sigmoid can be arbitrarily accurate [31]–[33]. In this paper, based on the approximation performance of BP network, a fusion network is proposed. After training, the proposed fusion networks can achieve the effects of the three classical fusion methods (weighted by matrices, by diagonal matrices and by scalars). The selection basis of the number of nodes and activation function in hidden layer nodes will also be given.

As shown in FIGURE 3, x_j $(j = 1, \cdots, M)$ are the inputs of the input layer; $w_{ij}^{(1)}$ $(i = 1, \cdots, q; \ j = 1, \cdots, M)$ are the weights between input layer and hidden layer, and $w_{ki}^{(2)}$ $(k = 1, \cdots, N; \ i = 1, \cdots, q)$ are the weights between hidden layer and output layer, respectively; $f(\cdot)$ and $g(\cdot)$ are the

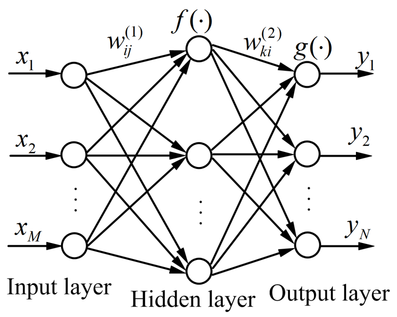


FIGURE 3. **The topology of BP network with single hidden layer.**

activation functions for the hidden and output layers, respectively; $\boldsymbol{d}(n) = [d_1, d_2, \cdots, d_N]$, $\boldsymbol{y}(n) = [y_1, y_2, \cdots, y_N]$ are the desired and actual outputs, respectively.

Using the approximation function of BP network to realize the classical fusion estimations involve the problems, such as the number of hidden layer nodes and the selection of activation functions.

First of all, noting that the fusion Equations (4), (9) and (14) are linear, linear functions are chosen as the activation functions. Such a choice can achieve the effects of classical fusion estimation and reduce the computational cost of network training and working. Next, some discussions about the number of nodes in hidden layer will be given.

Theorem 1: If a BP network with single hidden layer equivalent to the optimal fusion weighted by matrices in the sense of LMV through training, then the number of nodes in the hidden layer q must satisfy $q \geq n$ (n is the dimension of state).

Proof: Let $\hat{\boldsymbol{x}}_i(t|t)$ $(i = 1, \cdots, L)$ is local optimal unbiased estimates. According to the distributed fusion estimation framework shown as FIGURE 2 and the topology of BP network shown as FIGURE 3, the number of nodes in the input, hidden and output layers are M $(M = n \times L)$, q and n, respectively.

Both the activation function in hidden layer and the activation function in output layer are selected as linear transfer functions. The performance index for the network is:

$$\begin{aligned} E(\boldsymbol{w}) &= \frac{1}{2}\sum_{\gamma=1}^{Q}(\boldsymbol{d}_\gamma - \boldsymbol{y}_\gamma)^{\mathrm{T}}(\boldsymbol{d}_\gamma - \boldsymbol{y}_\gamma) \\ &= \frac{1}{2}\sum_{\gamma=1}^{Q}\boldsymbol{e}_\gamma^{\mathrm{T}}\boldsymbol{e}_\gamma \\ &= \frac{1}{2}\sum_{\gamma=1}^{Q}\sum_{k=1}^{n}e_{\gamma k}^2 \end{aligned} \tag{26}$$

where $\boldsymbol{w} \in \Re^{((M+n)\times q)\times 1}$ is the weight vector, Q is the number of training patterns, and the error for the γth input $\boldsymbol{e}_\gamma = \boldsymbol{d}_\gamma - \boldsymbol{y}_\gamma = \left[e_{11} \cdots e_{1n} \cdots e_{Q1} \cdots e_{Qn}\right]^{\mathrm{T}}$ ($\gamma = 1 \sim Q$), and vectors $\boldsymbol{e}_\gamma \in \Re^{N\times 1}$, $\boldsymbol{d}_\gamma \in \Re^{N\times 1}$, $\boldsymbol{y}_\gamma \in \Re^{N\times 1}$ $(N = Q \times n)$.

Find the minimum weight vector in performance index for the network of Equation (26) to get the system of Equations:

$$\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J}\boldsymbol{w} - \boldsymbol{J}^{\mathrm{T}}\bar{\boldsymbol{e}} = 0 \tag{27}$$

where the Jacobian matrix $\boldsymbol{J} \in \Re^{(Q\times n)\times U}$ and the error vector are defined as:

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial e_{11}}{\partial w_1} & \frac{\partial e_{11}}{\partial w_2} & \cdots & \frac{\partial e_{11}}{\partial w_U} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_{1n}}{\partial w_1} & \frac{\partial e_{1n}}{\partial w_2} & \cdots & \frac{\partial e_{1n}}{\partial w_U} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_{Q1}}{\partial w_1} & \frac{\partial e_{Q1}}{\partial w_2} & \cdots & \frac{\partial e_{Q1}}{\partial w_U} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_{Qn}}{\partial w_1} & \frac{\partial e_{Qn}}{\partial w_2} & \cdots & \frac{\partial e_{Qn}}{\partial w_U} \end{bmatrix}, \quad \bar{\boldsymbol{e}} = \begin{bmatrix} \boldsymbol{e}_{11} \\ \vdots \\ \boldsymbol{e}_{1n} \\ \vdots \\ \boldsymbol{e}_{Q1} \\ \vdots \\ \boldsymbol{e}_{Qn} \end{bmatrix} \tag{28}$$

If $\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J}$ is a non-singular matrix (Rank$(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J}) = (M+n)\times q$), we can get the least squares solution of the optimal weights $\boldsymbol{w}^*$ for the γth group of training as:

$$\boldsymbol{w}^* = (\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\boldsymbol{J}^{\mathrm{T}}\boldsymbol{e} \tag{29}$$

In order to prevent the matrix $\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J}$ singularity in Equation (29), LM (Levenberg-Marquardt) algorithm is used here, that is the Equation (29) is rewritten as:

$$\boldsymbol{w}^* = (\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J} + \mu\boldsymbol{I})^{-1}\boldsymbol{J}^{\mathrm{T}}\boldsymbol{e} \tag{30}$$

According to the BP network shown in Fig. 3, the output of the kth node of output layer is:

$$y_k = \sum_{i=1}^{q} w_{ki}^{(2)}\left(\sum_{j=1}^{M} w_{ij}^{(1)}x_j + \theta_i\right) + a_k \tag{31}$$

where the θ_i is the threshold of the ith node in hidden layer, and the a_k is the threshold of the kth node in output layer. Considering that fusion equations (4), (9) and (14) are linear homogeneous, the thresholds θ_i and a_k are set to zero. Then the output $\boldsymbol{y}_\gamma$ is:

$$\begin{aligned} \boldsymbol{y}_\gamma &= \begin{bmatrix} w_{11}^{(2)} & w_{12}^{(2)} & \cdots & w_{1q}^{(2)} \\ w_{21}^{(2)} & w_{22}^{(2)} & \cdots & w_{2q}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k1}^{(2)} & w_{k2}^{(2)} & \cdots & w_{kq}^{(2)} \end{bmatrix} \left(\begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & \cdots & w_{1M}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & \cdots & w_{2M}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{q1}^{(1)} & w_{q2}^{(1)} & \cdots & w_{qM}^{(1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \end{bmatrix} \right) \\ &= \boldsymbol{B}_{n\times M}\left[\left(\hat{\boldsymbol{x}}_{k|k}^{(1)}\right)^{\mathrm{T}} \quad \cdots \quad \left(\hat{\boldsymbol{x}}_{k|k}^{(L)}\right)^{\mathrm{T}}\right]_{1\times nL}^{\mathrm{T}} \end{aligned} \tag{32}$$

where matrix

$$\boldsymbol{B}_{n\times M} = \begin{bmatrix} w_{11}^{(2)} & \cdots & w_{1q}^{(2)} \\ \vdots & \ddots & \vdots \\ w_{k1}^{(2)} & \cdots & w_{kq}^{(2)} \end{bmatrix}_{n\times q} \begin{bmatrix} w_{11}^{(1)} & \cdots & w_{1M}^{(1)} \\ \vdots & \ddots & \vdots \\ w_{q1}^{(1)} & \cdots & w_{qM}^{(1)} \end{bmatrix}_{q\times M} = \boldsymbol{\Omega}\boldsymbol{\Lambda}$$

Rewrite the Equation (4) as:

$$\hat{\boldsymbol{x}}_{k|k}^{(M)} = \begin{bmatrix} \boldsymbol{A}_k^{(1)} & \cdots & \boldsymbol{A}_k^{(L)} \end{bmatrix}_{n\times nL} \begin{bmatrix} \left(\hat{\boldsymbol{x}}_{k|k}^{(1)}\right)^{\mathrm{T}} & \cdots & \left(\hat{\boldsymbol{x}}_{k|k}^{(L)}\right)^{\mathrm{T}} \end{bmatrix}_{1\times nL}^{\mathrm{T}} = \boldsymbol{A}_{n\times nL} \begin{bmatrix} \left(\hat{\boldsymbol{x}}_{k|k}^{(1)}\right)^{\mathrm{T}} & \cdots & \left(\hat{\boldsymbol{x}}_{k|k}^{(L)}\right)^{\mathrm{T}} \end{bmatrix}_{1\times nL}^{\mathrm{T}} \quad (33)$$

Comparing Equations (32) and (33), if the BP network is equivalent to the optimal fusion estimation weighted by matrices, then there must be

$$\boldsymbol{A}_{n\times nL} = \boldsymbol{B}_{n\times M} = \boldsymbol{\Omega}\boldsymbol{\Lambda} \quad (34)$$

From the full rank decomposition, when Rank($\boldsymbol{A}_{n\times nL}$) $= n$, there must be a row full matrix $\boldsymbol{\Omega} \in \Re^{n\times q}$ and a column full rank matrix $\boldsymbol{\Lambda} \in \Re^{q\times M}$ to make Equation (34) be founded, that is, $q \geq n$. The proof is completed.

Remark 1: From Theorem 1, it can be seen that if BP network with single hidden layer is used to achieve optimal fusion weighted matrices, the number of nodes q in the hidden layer is at least n. When $q < n$, the BP network will cause a part of states of the system to fail to achieve optimal fusion weighted by matrices. When $q > n$, some of the weight combinations in BP network are linearly related, which will increase the complexity of the network but have no effect. Theorem1 can improve the efficiency and reduce the unnecessary hidden nodes of the BP network.

Similar to Theorem 1, we can get the following deduction.

Deduction 1: If a BP network with single hidden layer equivalent to the optimal fusion weighted by diagonal matrices in the sense of LMV through training, then the number of nodes in the hidden layer q must satisfy $q \geq n$ (n is the dimension of state).

Remark 2: The accuracies of the three classical fusion algorithms are fusion weighted by matrices, by diagonal matrices and by scalars, respectively, from high to low. However, the minimum number of nodes in the hidden layer of the BP network has not changed, and it's just that some of the weights in the BP network are reduced to zero. That is to say, there is no difference in the efficiency of the three fusion algorithms based on BP network (BP net-based fusion weighted by matrices, by diagonal matrices and by scalar), but the accuracies and the training sets are different.

B. BP NET-BASED FUSION ALGORITHM TRAINED BY TRUE STATE

For BP net-based fusion algorithms trained by classical fusion algorithms, the process and measurement noise covariance matrices are needed to obtain the local estimations for BP network inputs, and cross-covariance matrices are also needed to obtain the fusion estimations for BP network training. However, in many cases the cross-covariance matrices are not easy to obtain or even impossible [39], which makes the training set of the BP network cannot be obtained and the network cannot complete the training. In this case, the true state can be use as the training set. Training BP networks with true state is widely used in various multi-sensor information fusion systems. However, there is on basis for the choice of activation functions and the number of nodes in hidden layer [20]–[26]. The following theorem will give some results about the BP net-based fusion algorithm trained by true state.

Theorem 2: If process noise $\boldsymbol{w}_k$ and measurement noises $\boldsymbol{v}_k^{(j)}$ $(j = 1, \cdots, L)$ are linearly related, the fusion BP network can reach the true state through training when the linear functions are selected as the activation functions.

Proof: The inputs of the fusion BP network are the local Klaman filters $\hat{\boldsymbol{x}}_{k|k}^{(j)}$ $(j = 1, \cdots, L$), which are linear combination of the initial state $\boldsymbol{x}_0$, process noises $\boldsymbol{w}_0 \sim \boldsymbol{w}_k$ and measurement noises $\boldsymbol{v}_0^{(j)} \sim \boldsymbol{v}_k^{(j)}$, that is

$$\hat{\boldsymbol{x}}_{k|k}^{(j)} = \boldsymbol{L}_k^{(j)}\left(\boldsymbol{x}_0,\ \boldsymbol{w}_0\cdots\boldsymbol{w}_k,\ \boldsymbol{v}_0^{(j)}\cdots\boldsymbol{v}_k^{(j)}\right) \quad (35)$$

where $\boldsymbol{L}_k^{(j)}(\cdot)$ represents linear transformation. The state $\boldsymbol{x}_k$ is a linear combination of the initial state $\boldsymbol{x}_0$ and process noises $\boldsymbol{w}_0 \sim \boldsymbol{w}_k$, that is

$$\boldsymbol{x}_k = \boldsymbol{L}_k^{(x)}\left(\boldsymbol{x}_0,\ \boldsymbol{w}_0\cdots\boldsymbol{w}_k\right) \quad (36)$$

where $\boldsymbol{L}_k^{(x)}(\cdot)$ represents linear transformation. Then the output $\boldsymbol{y}_\gamma$ in Equation (32) can be rewritten as:

$$\boldsymbol{y}_\gamma = \boldsymbol{L'}_k\left(\hat{\boldsymbol{x}}_{k|k}^{(1)},\ \cdots,\ \hat{\boldsymbol{x}}_{k|k}^{(L)}\right) = \boldsymbol{L''}_k(\boldsymbol{x}_0,\ \boldsymbol{w}_0\cdots\boldsymbol{w}_k,\ \boldsymbol{v}_0^{(1)}\cdots\boldsymbol{v}_k^{(1)},\ \cdots,\ \boldsymbol{v}_0^{(L)}\cdots\boldsymbol{v}_k^{(L)}) \quad (37)$$

where $\boldsymbol{L'}_k(\cdot)$ and $\boldsymbol{L''}_k(\cdot)$ represent linear transformations. Considering that the measurement noises are random, $\boldsymbol{L}_k^{(x)}$ and $\boldsymbol{L''}_k$ can be equivalent only when $\boldsymbol{v}_k^{(j)}$ $(j = 1, \cdots, L)$ can be linearly represented by $\boldsymbol{w}_k$. Considering that the inputs of the BP network are linear Kalman filters, so the activation functions of BP network should be linear. The proof is completed.

Deduction 2: If process noise $\boldsymbol{w}_k$ and measurement noises $\boldsymbol{v}_k^{(j)}$ $(j = 1, \cdots, L)$ have defined nonlinear relationships, the fusion BP network can be approximated to the true state when the nonlinear functions are selected as the activation functions.

Remark 3: If process noise $\boldsymbol{w}_k$ and measurement noises $\boldsymbol{v}_k^{(j)}$ $(j = 1, \cdots, L)$ are independent, the fusion BP network can not get a set of fixed weights. That is to say, when the measurement noise $\boldsymbol{v}_k^{(j)}$ $(j = 1, \cdots, L)$ which are used to generate the training set are different, the weights obtained by the training set are different.

C. RECURRENT NET-BASED FUSION ALGORITHM TRAINED BY CENTRALIZED FUSION ESTIMATIONS

For BP net-based fusion algorithms trained by classical fusion algorithms, many prior probabilities need to be known.

For BP net-based fusion algorithm trained by true state, the true state need to be known, and for different measurement noises, the weights of the network are different. Considering the limitations of the above two training methods, in order to learn the optimal result, we choose the training set as the centralized fusion result (centralized fusion is considered a global optimal estimate because there is no information loss).

Theorem 3: The recurrent network can approach the centralized fusion through training when the linear functions are selected as the activation functions.

Proof: The centralized fusion estimation $\hat{\boldsymbol{x}}_{k|k}^{(C)}$ is a linear combination of the initial state $\boldsymbol{x}_0$ and process noises $\boldsymbol{w}_0 \sim \boldsymbol{w}_k$ and measurement noises $\boldsymbol{v}_0^{(1)} \cdots \boldsymbol{v}_k^{(1)}, \ \cdots, \ \boldsymbol{v}_0^{(L)} \cdots \boldsymbol{v}_k^{(L)}$, that is

$$\hat{\boldsymbol{x}}_{k|k}^{(C)} = \boldsymbol{L}_k^{(C)}(\boldsymbol{x}_0, \ \boldsymbol{w}_0 \cdots \boldsymbol{w}_k, \ \boldsymbol{v}_0^{(1)} \cdots \boldsymbol{v}_k^{(1)}, \ \cdots, \ \boldsymbol{v}_0^{(L)} \cdots \boldsymbol{v}_k^{(L)}) \tag{38}$$

where $\boldsymbol{L}_k^{(C)}(\cdot)$ represents linear transformation. A linear recurrent network output $\boldsymbol{y}_\gamma$ is also a linear combination of inputs (here the inputs are $\hat{\boldsymbol{x}}_{k|k}^{(j)}$ $(j = 1, \cdots, L)$), so it can be written as:

$$\boldsymbol{y}_\gamma = \boldsymbol{L'}_k\left(\hat{\boldsymbol{x}}_{k|k}^{(1)}, \ \cdots, \ \hat{\boldsymbol{x}}_{k|k}^{(L)}\right) = \boldsymbol{L''}_k(\boldsymbol{x}_0, \ \boldsymbol{w}_0 \cdots \boldsymbol{w}_k, \ \boldsymbol{v}_0^{(1)} \cdots \boldsymbol{v}_k^{(1)}, \ \cdots, \ \boldsymbol{v}_0^{(L)} \cdots \boldsymbol{v}_k^{(L)}) \tag{39}$$

In theory, linear networks can learn this relationship, and considering the similarity of network structure, recurrent network is chosen here. The proof is completed.

Remark 4: Because centralized fusion is not limited to the minimum variance criterion and no information loss, its accuracy is higher than the classical three distributed fusion algorithms. The accuracy of the recurrent network obtained by learning centralized fusion is also higher than other networks.

V. SIMULATION EXAMPLE

We consider a tracking system with three sensors.

$$\boldsymbol{x}_{k+1} = \boldsymbol{\Phi}\boldsymbol{x}_k + \boldsymbol{\Gamma}\boldsymbol{w}_k \tag{40}$$

$$\boldsymbol{y}_k^{(j)} = \boldsymbol{H}_k^{(j)}\boldsymbol{x}_k + \boldsymbol{v}_k^{(j)}, j = 1, 2, 3 \tag{41}$$

where $\boldsymbol{x}_k = \begin{bmatrix} x_k & \dot{x}_k & y_k & \dot{y}_k \end{bmatrix}^{\mathrm{T}}$, x_k,$\dot{x}_k$, y_k, $\dot{y}_k$ are position and velocity on the $x-axes$ and $y-axes$ at time k. $\boldsymbol{y}_k^{(j)}$ $(j = 1, 2, 3)$ are the measurements, and

$$\boldsymbol{\Phi} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}, \boldsymbol{\Gamma} = \begin{bmatrix} 0.5T^2 & 0 \\ T & 0 \\ 0 & 0.5T^2 \\ 0 & T \end{bmatrix}$$

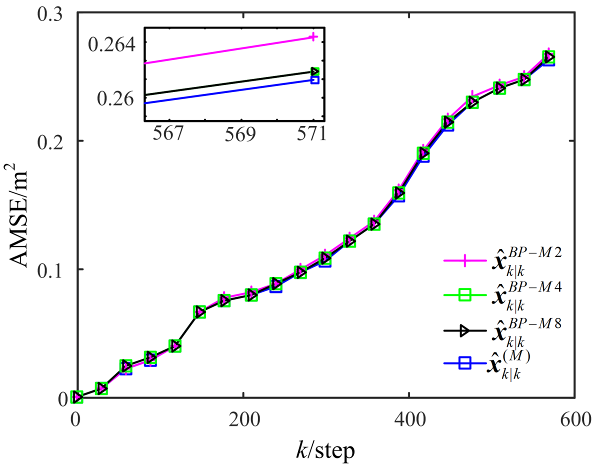


FIGURE 4. The AMSE of BP net-based matrix fusion.

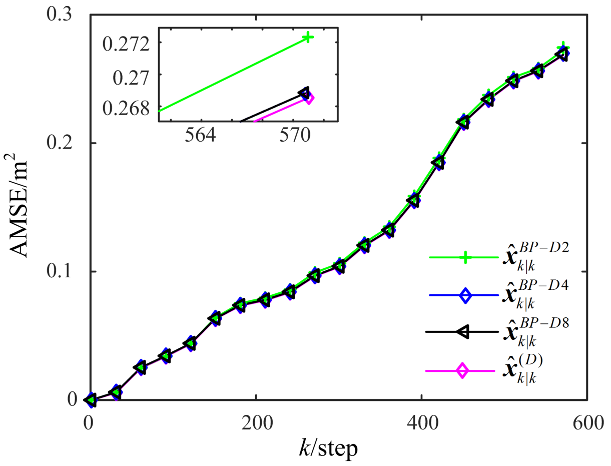


FIGURE 5. The AMSE of BP net-based diagonal matrix fusion.

$$\boldsymbol{H}_j = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad j = 1, 2, 3 \tag{42}$$

where T is the sampling period. $\boldsymbol{w}_k$ and $\boldsymbol{v}_k^{(j)}$ are independent Gaussian white noises with zero mean and variances $\boldsymbol{Q}_w$ and $\boldsymbol{R}^{(j)}$, respectively. In the simulation, $T = 0.1s$, $\boldsymbol{Q}_w = 1m^2/s^4$, $\boldsymbol{R}^{(1)} = \mathrm{diag}(0.33^2m^2/s^4, 0.38^2m^2/s^4)$ $\boldsymbol{R}^{(2)} = \mathrm{diag}(0.35^2m^2/s^4, 0.28^2m^2/s^4)$, $\boldsymbol{R}^{(3)} = \mathrm{diag}(0.31^2m^2/s^4, 0.22^2m^2/s^4)$; the initial value $\boldsymbol{x}_0 = \begin{bmatrix} 0m & 0m & 0m & 0m \end{bmatrix}^{\mathrm{T}}$, $\boldsymbol{P}_0 = \boldsymbol{I}_4$. The estimation performance is accumulated mean square error (AMSE) in position at time k [41]–[43]:

$$\mathrm{AMSE} = \sum_{t=0}^{k} \frac{1}{N} \sum_{i=1}^{N} \left((x_t^i - \hat{x}_{t|t}^i)^2 + (y_t^i - \hat{y}_{t|t}^i)^2\right) \tag{43}$$

where $(x_t^i, \ y_t^i)$ and $(\hat{x}_{t|t}^i, \ \hat{y}_{t|t}^i)$ are the true and estimated positions of the ith Monte Carlo experiment at time t.

First of all, according to Theorem 1, setting $f(\cdot)$ and $g(\cdot)$ are linear, $\theta_i = 0$ and $a_k = 0$ respectively. Then applying Lemma 1, Lemma 2, Lemma 3 and Lemma 4, respectively, we obtain the estimator $\hat{\boldsymbol{x}}_{k|k}^{(M)}$ weighted by matrices (the fusion

TABLE 1. The optimal fusion matrices.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.3307	2.1758×10^{-4}	0	0	0.2986	-0.0109	0	0	0.3707	0.0106	0	0
2	-2.4879×10^{-4}	0.3313	0	0	0.0124	0.2696	0	0	-0.0122	0.3991	0	0
3	0	0	0.1911	-0.0426	0	0	0.3101	0.0100	0	0	0.4988	0.0326
4	0	0	0.0564	0.0678	0	0	-0.0133	0.3391	0	0	-0.0431	0.5931

Columns 1 to 4, columns 5 to 8 and columns 9 to 12 are $\boldsymbol{A}_1(t|t)$, $\boldsymbol{A}_2(t|t)$ and $\boldsymbol{A}_3(t|t)$ of the classical fusion matrices in Equation (4), respectively.

TABLE 2. The network weight matrices of BP net-based fusion network with 2 hidden Layer nodes.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.3307	3.2740×10^{-4}	-3.6203×10^{-4}	6.6804×10^{-5}	0.2980	-0.0099	5.1737×10^{-4}	-0.0014	0.3713	0.0101	-1.5551×10^{-4}	-0.0013
2	1.6494×10^{-4}	1.9375×10^{-5}	0.0014	-3.0977×10^{-4}	1.4982×10^{-4}	1.0268×10^{-5}	0.0023	7.3897×10^{-5}	1.8649×10^{-4}	2.4416×10^{-5}	0.0037	2.4424×10^{-4}
3	-1.1428×10^{-4}	0.0026	0.1910	-0.0422	6.2513×10^{-5}	0.0021	0.3102	0.0102	5.1627×10^{-5}	0.0026	0.4987	0.0334
4	-8.8391×10^{-4}	2.5819×10^{-6}	2.5311×10^{-4}	-5.5924×10^{-5}	-7.9614×10^{-4}	2.9171×10^{-5}	4.0805×10^{-4}	1.7202×10^{-5}	-9.9204×10^{-4}	-2.3419×10^{-5}	6.5862×10^{-4}	4.7594×10^{-5}

Columns 1 to 12 are obtained by the weight vector $w^{(2)}_{4\times 2}w^{(1)}_{2\times 12}$ of BP net-based fusion network with 2 hidden Layer nodes.

TABLE 3. The network weight matrices of BP net-based fusion network with 4 hidden Layer nodes.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.3307	1.6356×10^{-4}	-2.1224×10^{-4}	2.5000×10^{-4}	0.2980	-0.0100	4.8183×10^{-4}	-5.0880×10^{-4}	0.3713	0.0099	-2.6960×10^{-4}	2.6364×10^{-4}
2	2.9657×10^{-7}	0.3309	-5.9072×10^{-4}	6.7161×10^{-4}	0.0103	0.2721	0.0014	0.0014	-0.0103	0.3970	-8.2966×10^{-4}	7.6829×10^{-4}
3	-1.1355×10^{-4}	1.9087×10^{-4}	0.1910	-0.0423	-1.3774×10^{-5}	8.0402×10^{-5}	0.3102	0.0097	1.2733×10^{-4}	-2.7056×10^{-4}	0.4988	0.0326
4	-5.2457×10^{-4}	7.6048×10^{-4}	0.0559	0.0687	4.2882×10^{-4}	-2.1316×10^{-4}	-0.0131	0.3384	9.6363×10^{-5}	-5.4619×10^{-4}	-0.0429	0.0529

Columns 1 to 12 are obtained by the weight vector $w^{(2)}_{4\times 4}w^{(1)}_{4\times 12}$ of BP net-based fusion network with 4 hidden Layer nodes.

TABLE 4. The network weight matrices of BP net-based fusion network with 8 hidden Layer nodes.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.3307	1.6356×10^{-4}	-2.1224×10^{-4}	2.5000×10^{-4}	0.2980	-0.0100	4.8183×10^{-4}	-5.0880×10^{-4}	0.3713	0.0099	-2.6960×10^{-4}	2.6364×10^{-4}
2	2.9653×10^{-7}	0.3309	-5.9072×10^{-4}	6.7161×10^{-4}	0.0103	0.2721	0.0014	0.0014	-0.0103	0.3970	-8.2966×10^{-4}	7.6829×10^{-4}
3	-1.1355×10^{-4}	1.9087×10^{-4}	0.1910	-0.0423	-1.3774×10^{-5}	8.0402×10^{-5}	0.3102	0.0097	1.2733×10^{-4}	-2.7056×10^{-4}	0.4988	0.0326
4	-5.2457×10^{-4}	7.6048×10^{-4}	0.0559	0.0687	4.2882×10^{-4}	-2.1316×10^{-4}	-0.0131	0.3384	9.6363×10^{-5}	-5.4619×10^{-4}	-0.0429	0.0529

Columns 1 to 12 are obtained by the weight vector $w^{(2)}_{4\times 8}w^{(1)}_{8\times 12}$ of BP net-based fusion network with 8 hidden Layer nodes.

TABLE 5. The network weight matrices of Elman net-based fusion network with 4 hidden Layer nodes.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.3156	9.6195×10^{-5}	0.0017	-1.9452×10^{-5}	0.3147	3.5408×10^{-4}	-0.0154	5.6878×10^{-5}	0.3697	2.7446×10^{-4}	0.0138	-3.4968×10^{-5}
2	-2.3041	0.3397	0.1712	-0.0018	2.2520	0.2578	-1.7646	0.0032	0.0507	0.4020	1.5924	-0.0013
3	0.0246	-2.6047×10^{-4}	0.1798	-2.4442×10^{-4}	0.1171	-0.0011	0.3420	-6.4192×10^{-5}	-0.1417	0.0013	0.4798	3.2771×10^{-4}
4	-0.0453	0.0058	6.5570	0.0678	0.4394	0.0223	-0.7854	0.3462	-0.3926	-0.0272	-5.7737	0.5985

Columns 1 to 12 are obtained by the weight vector $w^{(2)}_{4\times 4}w^{(1)}_{4\times 12}$ of the Elman net-based fusion network with 4 hidden Layer nodes.

TABLE 6. The network weight matrices of Elman net-based fusion network with 8 hidden Layer nodes.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.3075	1.7737×10^{-4}	0.0027	-2.5401×10^{-5}	0.3094	-3.0427×10^{-4}	-0.0180	6.9621×10^{-5}	0.3831	1.4796×10^{-4}	0.0153	-4.1184×10^{-5}
2	-2.3014	0.3396	0.1693	-0.0018	2.2669	0.2577	-1.7592	0.0032	0.0331	0.4021	1.5890	-0.0012
3	-0.0033	3.1552×10^{-5}	0.1896	-3.1610×10^{-4}	0.0072	2.4438×10^{-5}	0.3174	6.5293×10^{-5}	-0.0039	-5.4815×10^{-5}	0.4931	2.6972×10^{-4}
4	-0.0453	0.0058	6.5570	0.0679	0.4399	0.0223	-0.7849	0.3462	-0.3932	-0.0272	-5.7741	0.5985

Columns 1 to 12 are obtained by the weight vector $w^{(2)}_{4\times 8}w^{(1)}_{8\times 12}$ of the Elman net-based fusion network with 8 hidden Layer nodes.

TABLE 7. The network weight matrices of BP net-based fusion network with 4 hidden Layer nodes.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.2946	0.0149	0.0032	0.0212	0.3815	-0.0147	-0.0104	-0.0089	0.3869	0.0029	0.0072	-0.0129
2	-0.0423	0.3496	-0.0873	0.1018	0.0577	0.2658	-0.0554	-0.0161	-0.0155	0.3856	0.1482	-0.0838
3	-0.0140	0.0135	0.2025	-0.0542	-0.0080	0.0073	0.2854	0.0435	0.0220	-0.0210	0.5121	0.0111
4	-0.0442	0.0387	0.1212	0.0095	0.0105	-0.0262	-0.0972	0.4071	0.0337	-0.0131	-0.0240	0.5743

Columns 1 to 12 are obtained by the weight vector $w^{(2)}_{4\times 4}w^{(1)}_{4\times 12}$ of BP net-Based fusion network with 4 hidden Layer nodes.

TABLE 8. The network weight matrices of BP net-based fusion network with 4 hidden layer nodes.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.2890	0.0045	-0.0263	0.0361	0.2332	0.0512	-0.0335	-0.0017	0.4777	-0.0554	0.0598	-0.0321
2	-0.0869	0.03329	-0.1265	0.1407	-0.1058	0.3638	-0.0115	-0.0843	0.1926	0.3048	0.1381	-0.0548
3	-0.0347	0.0254	0.1949	-0.0415	0.0236	-0.0124	0.2767	0.0333	0.0111	-0.0127	0.5284	0.0087
4	-0.1111	0.0948	0.0132	0.1141	0.0837	-0.0594	-0.0450	0.3545	0.0274	-0.0364	-0.0318	0.5229

Columns 1 to 12 are obtained by the weight vector $w^{(2)}_{4\times 4}w^{(1)}_{4\times 12}$ of BP net-Based fusion network with 4 hidden Layer nodes.

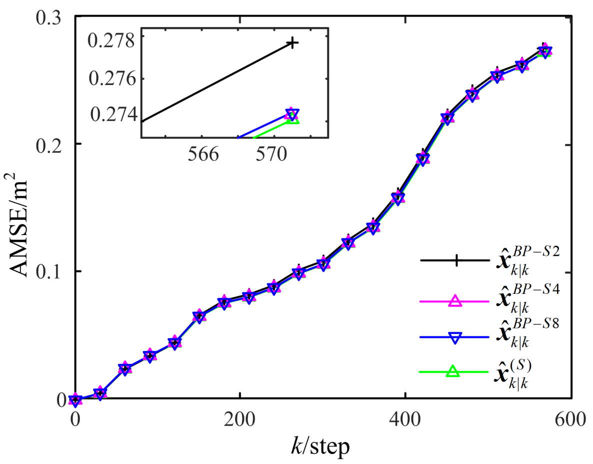


FIGURE 6. The AMSE of BP net-based scalar fusion.

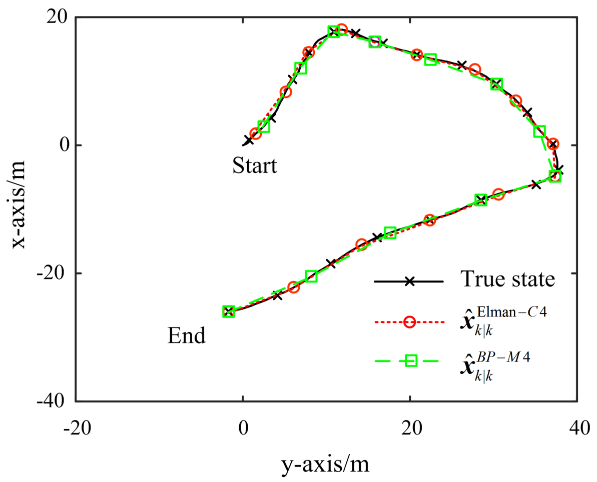


FIGURE 8. The track trajectories of BP net-based fusion and Elman net-based fusion.

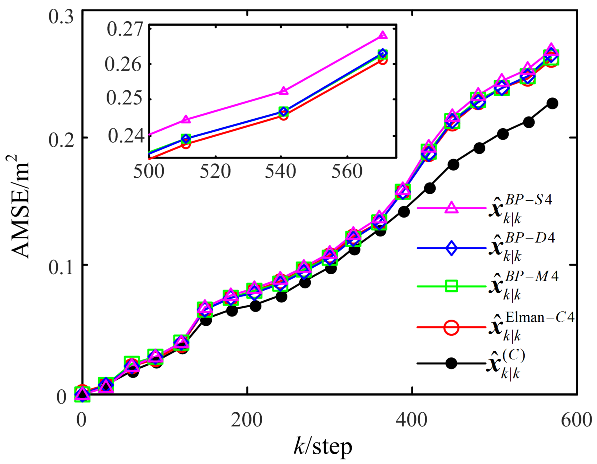


FIGURE 7. The AMSEs of three BP net-based fusions and Elman net-based fusion.

matrices, please see TABLE 1), the $\hat{x}^{(S)}_{k|k}$ weighted by scalars, the estimator $\hat{x}^{(D)}_{k|k}$ weighted by diagonal matrices, and centralized fusion estimator $\hat{x}^{(C)}_{k|k}$. Then, using local estimation $\hat{x}^{(j)}_{k|k}$ $(j = 1, 2, 3)$ as BP network inputs, and three weighted fusion estimations as training sets, three fusion networks are built. The AMSE of the fusion estimations are shown in FIGURE 4-6, where $\hat{x}^{BP-\alpha\beta}_{k|k}$, α means the fusion method (M means matrix fusion, S means scalars fusion and D means diagonal matrix fusion), and β means the number of hidden layer nodes. In simulation, 50 Monte Carlo experiments are performed for 600-step test. From FIGURE 4-6, the accuracies of the BP networks with 2 hidden nodes are lowest in position (Because there are fewer than 4 nodes in the hidden layer, the fusion coefficients of the velocity state components are not learned, please see TABLE 2). The accuracies of the BP networks with 4 hidden nodes and 8 hidden nodes are the same and the network weight matrices are almost the same as the three classical fusion methods (Please see TABLES 3-4), and it verifies Theorem 1. From FIGURE 7, we can see that the accuracies of the three BP net-based fusion algorithm from high to low are $\hat{x}^{BP-M4}_{k|k}$, $\hat{x}^{BP-D4}_{k|k}$ and $\hat{x}^{BP-S4}_{k|k}$, respectively.

Based on recurrent network Elman, another fusion algorithm trained by centralized fusion estimations (Elman net-Based fusion algorithm) is proposed and simulated. Here, an Elman network with 4 hidden layer nodes ($\hat{x}^{Elman-C4}_{k|k}$) is chosen for simulation. From FIGURE 7, we can see that the accuracy of $\hat{x}^{Elman-C4}_{k|k}$ is higher than $\hat{x}^{BP-M4}_{k|k}$, $\hat{x}^{BP-D4}_{k|k}$ and $\hat{x}^{BP-S4}_{k|k}$, and it verifies Theorem 3. In addition, it can be seen from TABLE 5-6 that the weights of Elman net-based fusion network are convergent, and the weights remain basically unchanged as the number of nodes in the hidden layer of the network increases. The track trajectories of various fusion algorithms are shown in FIGURE 8.

The network weight matrices of the distributed fusion algorithm based on BP network by using local estimations as inputs and true states as training set are given in

TABLES 7 and 8. From the two tables, due to the different training sets, the network weights have changed significantly, and the network has non-convergence for different training sets. It verifies Theorem 2.

VI. CONCLUSION

Two types of multi-sensor distributed fusion frameworks based on ANN and machine learning are proposed. One is based on BP network, and the other is based on Elman network. The main works are the following:

(1) Using local estimations as the inputs and estimations of three classic distributed fusions as the training sets, three distributed fusion algorithms based on BP network are proposed. In addition, the selection basis of the number of nodes in hidden layer is given. The basis can improve the efficiency and reduce the unnecessary hidden nodes of the network. For these fusion algorithms, the process and measurement noise covariance matrices and cross-covariance matrices are needed, which will increase the limitations of the algorithm.

(2) The BP net-based fusion algorithm which uses local estimations as inputs and true states as training set is analyzed. The unavailability of the true states and the instability of weights for different training sets are the limitations of the algorithm.

(3) By using local estimations as inputs and centralized fusion estimation as training set, Elman net-based distributed fusion algorithm is proposed, in the case that neither true states nor cross-covariance matrices is available. This method is not limited to the minimum variance criterion, so its accuracy is higher than the classical three distributed fusion algorithms.

REFERENCES

[1] S. Sun, H. Lin, J. Ma, and X. Li, "Multi-sensor distributed fusion estimation with applications in networked systems: A review paper," *Inf. Fusion*, vol. 38, pp. 122–134, Nov. 2017.

[2] X. R. Li, Y. Zhu, J. Wang, and C. Han, "Optimal linear estimation fusion. I. Unified fusion rules," *IEEE Trans. Inf. Theory*, vol. 49, no. 9, pp. 2192–2208, Sep. 2003.

[3] Q. Gan and C. J. Harris, "Comparison of two measurement fusion methods for Kalman-filter-based multisensor data fusion," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 37, no. 1, pp. 273–279, Jan. 2001.

[4] J. A. Roecker and C. D. McGillem, "Comparison of two-sensor tracking methods based on state vector fusion and measurement fusion," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 24, no. 4, pp. 447–449, Jul. 1988.

[5] H. R. Hashemipour, S. Roy, and A. J. Laub, "Decentralized structures for parallel Kalman filtering," *IEEE Trans. Autom. Control*, vol. AC-33, no. 1, pp. 88–94, Jan. 1988.

[6] B. S. Paik and J. H. Oh, "Gain fusion algorithm for decentralised parallel Kalman filters," *IEE Proc.-Control Theory Appl.*, vol. 147, no. 1, pp. 97–103, Jan. 2000.

[7] N. A. Carlson and J. H. Oh, "Comments, with reply, on federated square root filter for decentralized parallel processes," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 27, no. 6, pp. 946–951, Nov. 1991.

[8] N. A. Carlson, "Federated filter for fault-tolerant integrated navigation systems," in *Proc. Position Location Navigat. Symp., Rec. Navigat. 21st Century (PLANS)*, Orlando, FL, USA, Jan. 2003, pp. 110–119.

[9] S. Roy and R. A. Iltis, "Decentralized linear estimation in correlated measurement noise," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 27, no. 6, pp. 939–941, Nov. 1991.

[10] K. H. Kim, "Development of track to track fusion algorithms," in *Proc. Amer. Control Conf. (ACC)*, Baltimore, MD, USA, Aug. 2005, pp. 1037–1041.

[11] S.-L. Sun and Z.-L. Deng, "Multi-sensor optimal information fusion Kalman filter," *Automatica*, vol. 40, no. 6, pp. 1017–1023, Jun. 2004.

[12] S.-L. Sun, "Multi-sensor information fusion white noise filter weighted by scalars based on Kalman predictor," *Automatica*, vol. 40, no. 8, pp. 1447–1453, Aug. 2004.

[13] Z.-L. Deng, Y. Gao, L. Mao, Y. Li, and G. Hao, "New approach to information fusion steady-state Kalman filtering," *Automatica*, vol. 41, no. 10, pp. 1695–1707, Oct. 2005.

[14] B. Widrow and M. A. Lehr, "30 years of adaptive neural networks: Perceptron, Madaline, and backpropagation," *Proc. IEEE*, vol. 78, no. 9, pp. 1415–1442, Sep. 1990.

[15] E. Oja, "Simplified neuron model as a principal component analyzer," *J. Math. Biol.*, vol. 15, no. 3, pp. 267–273, Nov. 1982.

[16] F. Amoozegar, "Neural network fusion capabilities for efficient implementation of tracking algorithms," *Opt. Eng*, vol. 36, no. 3, p. 692, Mar. 1997.

[17] J. Zhongliang, X. Hong, and Z. Xueqin, "Information fusion and tracking of maneuvering targets with artificial neural network," in *Proc. IEEE Int. Conf. Neural Netw. (ICNN)*, Orlando, FL, USA, Dec. 2002, pp. 3403–3408.

[18] Y. Kuai, G. Wen, and D. Li, "Learning fully convolutional network for visual tracking with multi-layer feature fusion," *IEEE Access*, vol. 7, pp. 25915–25923, 2019.

[19] J. Ghosh and R. L. Holmberg, "Multisensor fusion using neural networks," in *Proc. 2nd IEEE Symp. Parallel Distrib. Process.*, Dallas, TX, USA, Dec. 2002, pp. 812–815.

[20] F. N. Chowdhury, "A neural approach to data fusion," in *Proc. Amer. Control Conf. (ACC)*, Seattle, WA, USA, Aug. 2005, pp. 1693–1697.

[21] L.-W. Fong and C.-Y. Fan, "Multisensor fusion algorithms for maneuvering target tracking," in *Proc. 1st IEEE Int. Conf. E-Learn. Ind. Electron.*, Hammamet, Tunisia, Dec. 2006, pp. 80–84.

[22] N. S. V. Rao, "Nadaraya-Watson estimator for sensor fusion problems," in *Proc. Int. Conf. Robot. Autom.*, Albuquerque, NM, USA, Nov. 2002, pp. 2069–2074.

[23] N. S. V. Rao, "Fusion methods in multiple sensor systems using feedforward sigmoid neural networks," *Intell. Automat. Soft Comput.*, vol. 5, no. 1, pp. 21–30, Jan. 1999.

[24] N. S. V. Rao, "Nearest neighbor projective fuser for function estimation," in *Proc. 5th Int. Conf. Inf. Fusion (FUSION)* Annapolis, MD, USA, 2002, pp. 1154–1161.

[25] Q. Liu, X. Wang, and N. S. V. Rao, "Artificial neural networks for estimation and fusion in long-haul sensor networks," in *Proc. 18th Int. Conf. Inf. Fusion (Fusion)*, Washington, DC, USA, 2015, pp. 460–467.

[26] K. Brigham, B. V. K. V. Kumar, and N. S. V. Rao, "Learning-based approaches to nonlinear multisensor fusion in target tracking," in *Proc. 16th Int. Conf. Inf. Fusion*, Istanbul, Turkey, 2013, pp. 1320–1327.

[27] P. J. Werbos, "New tools for prediction and analysis in the behavioral sciences," Ph.D. dissertation, Harvard Univ., Cambridge, MA, USA, 1974.

[28] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.

[29] P. J. Werbos, "Neurocontrol and elastic fuzzy logic: Capabilities, concepts, and applications," *IEEE Trans. Ind. Electron.*, vol. 40, no. 2, pp. 170–180, Apr. 1993.

[30] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[31] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control, Signals Syst.*, vol. 2, no. 4, pp. 303–314, Dec. 1989.

[32] K.-I. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Netw.*, vol. 2, no. 3, pp. 183–192, Jan. 1989.

[33] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, Jan. 1989.

[34] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci. USAA*, vol. 79, no. 8, pp. 2554–2558, Apr. 1982.

[35] J. L. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.

[36] A. Van Ooyen and B. Nienhuis, "Improving the convergence of the back-propagation algorithm," *Neural Netw.*, vol. 5, no. 3, pp. 465–471, Jan. 1992.

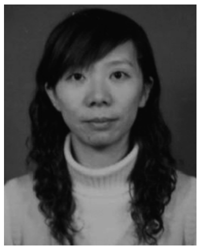
[37] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Trans. Neural Netw.*, vol. 5, no. 6, pp. 989–993, Nov. 1994.

[38] J. S. Meditch, *Stochastic Optimal Linear Estimation and Control*. New York, NY, USA: McGraw-Hill, 1969, pp. 304–348.

[39] S. C. Kremer, "On the computational power of Elman-style recurrent networks," *IEEE Trans. Neural Netw.*, vol. 6, no. 4, pp. 1000–1004, Jul. 1995.
[40] H. M. Lynn, S. B. Pan, and P. Kim, "A deep bidirectional GRU network model for biometric electrocardiogram classification based on recurrent neural networks," *IEEE Access*, vol. 7, pp. 145395–145405, 2019.
[41] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans. Autom. Control*, vol. 54, no. 6, pp. 1254–1269, Jun. 2009.
[42] G. Hao, S.-L. Sun, and Y. Li, "Nonlinear weighted measurement fusion Unscented Kalman Filter with asymptotic optimality," *Inf. Sci.*, vol. 299, pp. 85–98, Apr. 2015.
[43] G. Hao and S. Sun, "Distributed fusion cubature Kalman filters for nonlinear systems," *Int. J. Robust. Nonlinear Control*, vol. 29, no. 17, pp. 5979–5991, Nov. 2019.

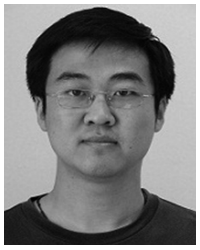
ZHENGXIAO PENG received the B.S. degree in automation from the School of Electrical and Information Engineering, Ma'anshan, China, in 2016. He is currently pursuing the M.S. degree in control science and engineering with Heilongjiang University, China.

His research interest includes information fusion, state estimation for nonlinear systems, and artificial intelligence.



YUN LI was born in Qiqihar, Heilongjiang, China, in 1978. She received the B.S. and M.S. degrees in automation and the Ph.D. degree in control science and engineering from Heilongjiang University, Harbin, in 2005 and 2018, respectively.

Since 2005, she has been an Assistant Professor with the Computer and Information Engineering Department, Harbin University of Commerce. She is the author of seven books, more than 30 articles, and two inventions. Her research interests include information fusion and state estimation.



GANG HAO was born in Harbin, Heilongjiang, China, in 1980. He received the B.S. and M.S. degrees in automation from Heilongjiang University, Harbin, in 2006, and the Ph.D. degree in control science and engineering from Harbin Engineering University, Harbin, in 2011.

Since 2006, he has been an Assistant Professor with the Electronic Engineering Department, Heilongjiang University. He is the author of a book, more than 50 articles, and more than ten inventions. His research interests include information fusion, state estimation for nonlinear system, and artificial intelligence. He is also the Director of the artificial intelligence society of Heilongjiang province.