Hierarchical Fuzzy Neural Networks With Privacy Preservation for Heterogeneous Big Data

Article *in* IEEE Transactions on Fuzzy Systems · September 2020

DOI: 10.1109/TFUZZ.2020.3021713

4 authors, including:

Ye Shi
University of Technology Sydney
19 PUBLICATIONS **69** CITATIONS
SEE PROFILE

Yu-Cheng Chang
12 PUBLICATIONS **212** CITATIONS
SEE PROFILE

Chin-Teng Lin
National Chiao Tung University
601 PUBLICATIONS **13,173** CITATIONS
SEE PROFILE

Hierarchical fuzzy neural networks with privacy preservation for heterogeneous big data

Leijie Zhang, Ye Shi, *Member, IEEE*, Yu-Cheng Chang, and Chin-Teng Lin *Fellow, IEEE*

***Abstract*—Heterogeneous big data poses many challenges in machine learning. Its enormous scale, high dimensionality, and inherent uncertainty make almost every aspect of machine learning difficult, from providing enough processing power to maintaining model accuracy to protecting privacy. However, perhaps the most imposing problem is that big data is often interspersed with sensitive personal data. Hence, we propose a privacy-preserving hierarchical fuzzy neural network (PP-HFNN) to address these technical challenges while also alleviating privacy concerns. The network is trained with a two-stage optimization algorithm, and the parameters at low levels of the hierarchy are learned with a scheme based on the well-known alternating direction method of multipliers, which does not reveal local data to other agents. Coordination at high levels of the hierarchy is handled by the alternating optimization method, which converges very quickly. The entire training procedure is scalable, fast and does not suffer from gradient vanishing problems like the methods based on back-propagation. Comprehensive simulations conducted on both regression and classification tasks demonstrate the effectiveness of the proposed model. Our code is available online[1].**

***Index Terms*—Hierarchical fuzzy neural networks, heterogeneous big data, distributed clustering, privacy preservation, alternating optimization**

I. Introduction

Big data and its benefits have become a ubiquitous part of research in many areas, including social networks, the Internet of Things, commerce, astronomy, biology, medicine, and more [1], [2]. However, with increasing regularity, the gains being made seem to come at the threat of violations to the privacy and security of our personal data [3]. This issue is attracting serious attention in both society and the research community, especially post the Facebook data privacy scandal [4]. In addition to privacy concerns, heterogeneous big data and its inherent uncertainty, enormous scales and high dimensionality are posing even more challenges to traditional machine learning methods —- for instance, the extraordinary amounts of computing power needed to process big data and the negative effects uncertainty can have on a model's accuracy and performance [5], [6]. We need novel and efficient machine learning techniques to cope with all these challenges.

Usually, data heterogeneity stems from the nature of the information to be captured and/or the methods used to generate or acquire the data, and it can exist at either the sample, the feature level or both. Traditional data processing methods, such as data cleaning, data integration, dimension reduction and data normalization, may need to be applied in combination to ensure they effectively reduce data disparity [7]. In addition to heterogeneity, uncertainty is another problematic characteristic of heterogeneous big data. Loosely defined as "how well the data speaks to the question of interest with respect to the model" [8], uncertainty over whether the available data will meet the demands of the task begins at the collection stage. Neither sensors nor humans are immune to measurement errors; qualitative data, such as social media accounts, are often subjective in nature; and the data collected may not be relevant or might be incomplete. Currently, a powerful and effective solution for dealing with uncertainty is to build the model through a fuzzy neural network (FNN), which involves fuzzy rules and fuzzy inference systems [9], [10], to make optimal use of the given data [11], [12]. FNNs deal with data uncertainty through 'fuzzification' operations and architectures based on if-then rules. However, in a standard fuzzy system, the number of fuzzy rules increases exponentially with the number of system variables [13]–[15]. This limits the feasibility of applying standard FNNs to heterogeneous big data. In an attempt to overcome the rule explosion problem, Raju, as far back as 1991, proposed the first hierarchical fuzzy system [13]. It consisted of a number of low-dimensional fuzzy systems connected in a hierarchical structure such that the total number of rules increased linearly with the number of input variables. In the years since then, hierarchical fuzzy systems have been studied in depth [15]–[18] and applied to many practical problems, such as price negotiation [19], video de-interlacing [20], linguistic attribute hierarchy [21], weapon target assignment [22], cloud resources demand prediction [23], and more. However, there is very little literature on using hierarchical fuzzy neural network (HFNN) to address the uncertainty, privacy concerns and computational overhead associated with big data.

As one of the most salient problems in data science, privacy-preserving machine learning is attracting attention [24]–[27]. Generally, there are two types of privacy-preserving machine learning methods; one relies on perturbation and randomization, the other on segmenting the data. Perturba-

Leijie Zhang and Ye Shi contributed equally to this work. This work was supported in part by the Australian Research Council (ARC) under discovery grant DP180100670 and DP180100656. Research was also sponsored in part by the Australia Defence Innovation Hub under Contract No. P18-650825, and US Office of Naval Research Global under Cooperative Agreement Number ONRG-NICOP-N62909-19-1-2058. We also thank the NSW Defence Innovation Network and NSW State Government of Australia for financial support in part of this research through grant DINPP2019 S1-03/09. (Corresponding author: Ye Shi.)

Leijie Zhang, Ye Shi, Yu-Cheng Chang and Chin-Teng Lin are with CIBCI lab, Australian Artificial Intelligence Institute, the School of Computer Science, University of Technology, Sydney, NSW 2007, Australia. Email: Leijie.Zhang@student.uts.edu.au, Ye.Shi-1@uts.edu.au, Yu-Cheng.Chang@student.uts.edu.au, Chin-Teng.Lin@uts.edu.au.

[1]https://github.com/leijiezhang/PP_HFNN

tion/randomization methods alter the data before releasing or sharing them [25], which offers some limited privacy protection at the sacrifice of some performance. Methods based on segmentation typically distribute the data among multiple agents. Neighboring agents cooperate with each other to learn global results but without revealing their individual data to others [28]. Distributed machine learning algorithms have emerged out of these multi-agent data sharing scenarios as a solution to preserving privacy [29]–[33]. Distributed algorithms are well suited to big data environments given there are limits on the amount of data a single agent or a centralized algorithm can feasibly handle. To harness the communications overhead and storage capacity needed to process information at the big data scale, the data needs to be distributed throughout a network of interconnected agents [34].

This work brings the advantages of both hierarchical and distributed architectures to an FNN to tackle the privacy, uncertainty and feasibility constraints associated with heterogeneous big data. HFNNs are specifically designed to extract features from heterogeneous data and to reduce the number of fuzzy rules without compromising performance, which helps to overcome the potential exponential rule explosion. The system variables of the heterogeneous data are divided into several independent subsets, each of which only consists of homogeneous data that is then associated with an FNN at the lower levels of the hierarchy. The outputs of the low-level FNNs are coordinated at the higher levels of the hierarchy. For the purposes of this paper, we selected the alternating optimization (AO) method for high-level coordination. This is a widely-used method that sequentially optimizes the objective function over one variable while fixing the other [35]–[37]. Compared to gradient-based algorithms, AO has several merits: (i) At each iteration, AO divides a difficult problem into much easier sub-problems, which usually have a closed-form solution; (ii) AO is easy to implement as there is no need to tune the optimization parameters, like the learning rate or step sizes; and (iii) it converges very quickly in practice and does not suffer from the vanishing gradient problem associated with gradient-based learning methods. Further, each low-level FNN operates in a distributed computation framework to combat the challenges raised by numerous data samples as well as to alleviate privacy concerns. More specifically, a distributed clustering method, built on the well-known alternating direction method of multipliers (ADMM), locates the parameters in the antecedent layer of each low-level FNN. ADMM has proven to be an efficient solution for distributed computation problems [38], [39]. We explored many algorithms designed for HFNNs but, to our best knowledge, an HFNN method with privacy preservation for heterogeneous big data has not yet been theorized or developed. Therefore, the contributions of this paper are three-fold:

- We propose a new privacy-preserving HFNN model (PP-HFNN) to address some of the inherent issues with combining big heterogeneous data with machine learning tasks: massive scales, heterogeneity, uncertainty, and privacy concerns.
- Privacy is preserved by using a distributed clustering method to identify the parameters in the antecedent layer of each low-level FNN. This approach is also good at handling massive amounts of data.
- We have also developed a two-stage optimization algorithm to train the HFNN. Parameter learning for the low-level FNNs is based on the well-known ADMM, while coordination at the higher levels of the hierarchy relies on the AO method. The algorithm is scalable, fast, and does not suffer gradient vanishing problems, as is the case with methods based on back-propagation.

The remainder of the paper is organized as follows. In Section II, we discuss relevant works on heterogeneous data, hierarchical fuzzy neural networks and distributed machine learning algorithms. Section III is devoted to the PP-HFNN modeling. In Section IV, we present the two-stage optimization algorithm, which consists of a distributed clustering method for the low-level FNNs and an AO method for high-level coordination. The results of a comprehensive simulation for both regression and classification tasks are presented and discussed in Section V to confirm the effectiveness of the proposed privacy-preserving HFNN model, and Section VI concludes the paper.

II. Related Work

This work is the first to combine privacy preservation with an HFNN designed for heterogeneous big data and a distributed computing environment. Therefore, in this literature review, we briefly cover relevant work in the three critical elements of PP-HFNN: heterogeneous data, HFNNs and distributed machine learning algorithms.

A. Heterogeneous data

As a way to resolve conflicts in heterogeneous data merged from a variety of sources, Li et al. proposed an optimization framework to minimize the overall deviation between ground truths and the observations provided [40]. Zhang et al.'s solution combined principal component analysis (PCA) to reduce the number of dimensions with correlation analysis to investigate the interactive relations between features [41]. However, important information can be lost with this strategy, as PCA and correlation analysis do not consider nonlinear representative and interactive relations between features. A method based on support vector machine (SVM) was developed by Lewis et al. to infer functional gene annotations from heterogeneous data comprising protein sequences and structures [42]. Chen et al. developed a stacked denoising autoencoder to learn feature representations from hierarchical human mobility data. With this approach, they were able to predict the risk of traffic accidents [43]. Zuo et al. [44] proposed a fuzzy heterogeneous method to address domain adaptation with heterogeneous data spaces. However, none of the above methods [40]–[44] are a suitable solution to the uncertainty, scale issues and the privacy concerns associated with big data environments.

NOMENCLATURE

Abbreviations

ADMM	Alternating direction method of multipliers.
AO	Alternating optmization.
FNN	Fuzzy neural network.
HFNN	Hierarchical fuzzy neural network.
PP-HFNN	Privacy-preserving hierarchical fuzzy neural network.

Notations

$\mathbf{m}, \sigma$	Notation for the parameters in the low level of the hierarchy.
$\mathbf{w}, \mathbf{v}$	Notation for the parameters in the high level of the hierarchy.
$A_{r,j}^{b}$	The j-th fuzzy set of the r-th rule on the b-th low-level FNN.
b	Notation for the low-level FNN.
i	Notation for the sample.
j	Notation for the feature.
k	Notation for the cluster and the fuzzy rule.
l	Notation for the agent.
$x_{i,j}^{b,k}$	The j-th component of the i-th sample assigned to the k-th agent of the b-th low-level FNN.

Sets

$\mathcal{C}_k^b$	The k-th cluster in the b-th low level FNN.
$\mathcal{F}$	The set of features.
$\mathcal{F}_b$	The subset of features of the b-th low level FNN.
$\mathcal{N}$	The set of samples.
$\mathcal{N}_{b,k}$	The subset of training data assigned to the k-th agent of the b-th low-level FNN.

B. Hierarchical fuzzy neural networks

Over the past decade, a range of hierarchical structures of FNN have been studied for their benefits to different applications [45]–[48]. A hierarchical singleton-type recurrent FNN was proposed in [45] for noisy speech recognition, consisting of two singleton-type recurrent FNNs: one responsible for noise filtering and the other for recognition. Mohammadzadeh et al. [46] proposed a hierarchical interval Type-2 FNN designed to synchronize uncertain chaotic systems. Searching for a supervised framework more resilient to noisy inputs than the classical non-fuzzy neural network, Krleža et al. [47] proposed a fuzzy graph neural network based on a combination of fuzzy logic and a recursive neural network for graph-matching tasks. Deng et al. [48] developed a hierarchical network structure by fusing a conventional FNN and a deep neural network. Benefiting from fuzzy logic, this method is able to increase classification accuracy with data that contains a high level of uncertainty. However, these hierarchical FNNs [45]–[48] cannot handle heterogeneous big data effectively or efficiently, and some still suffer from the curse of dimensionality concerning fuzzy rules. Unlike these approaches, our PP-HFNN framework is designed to extract features from heterogeneous data and reduce the number of fuzzy rules without compromising performance.

C. Distributed machine learning algorithms

Distributed machine learning algorithms have been widely investigated [29]–[33]. For instance, Forero et al. [29] developed a consensus-based distributed SVM for a multi-agent network, where the training data is distributed across different agents but data exchange among agents is prohibited. Qin et al. [30] proposed a distributed K-means algorithm and a distributed fuzzy c-means model for wireless sensor networks based on distributed consensus strategies. A distributed extreme learning machine with kernels based on MapReduce was proposed by Zhao et al. [31], designed for very fast learning speeds. Ye et al. [32] developed a decentralized multi-task extreme learning machine based on a hybrid Jacobian and Gauss-Seidel proximal multi-block ADMM method. A decentralized training algorithm for Echo State Networks was developed by Scardapane et al. [33] directed toward distributed big data applications. But, although these distributed machine learning algorithms [29]–[33] can reduce computational requirements and alleviate potential privacy concerns, they may not be effective strategies for dealing with data uncertainty and heterogeneity.

In the last few years, several distributed algorithms for FNN have been proposed [49]–[51]. Fierimonte et al. presented a decentralized algorithm for a random-weight FNN, where the parameters in the antecedent layer are randomly selected instead of being estimated [49]. Subsequent work saw an online implementation of the same FNN structure in [50]. However, a random method of identifying parameters may result in very large deviations of accuracy during the learning process. In addition, this approach suffers from the curse of dimensionality because the number of fuzzy rules increases exponentially as the input space increases. Moreover, the distributed algorithms are only applied to the consequent layers of the FNN. To retrieve the parameters of the antecedent layer, all the data must first be processed by a central agent. A further issue is that this method does not provide any level of privacy protection, and the FNNs proposed can neither handle the scale of big data nor any heterogeneity issues. Recently, a consensus learning method for distributed FNN was proposed in [51], where a distributed clustering method and a distributed parameter learning method were respectively developed to locate the parameters in the antecedent and consequent layers of an FNN. Yet, again, none of the above distributed algorithms for FNN [49]–[51] can address the issue of data heterogeneity in big data environments.

III. MODEL FORMULATION OF THE PP-HFNN FOR HETEROGENEOUS BIG DATA

Heterogeneous big data is inherently characterized by its enormous scale, heterogeneity and uncertainty and is often interspersed with sensitive personal data. HFNN, which takes advantage of the hierarchical structure and fuzzy if-then rules, is effective for tackling the challenges of data heterogeneity and uncertainty. But the issues of enormous scale and data privacy fall into the domain of distributed privacy-preserving algorithms. As shown in Fig. 1, these challenges motivated our PP-HFNN model as a promising solution.

PP-HFNN consists of a two-level hierarchical structure. The lower level contains multiple FNNs, and the outputs of these FNNs are in the higher level. During the data

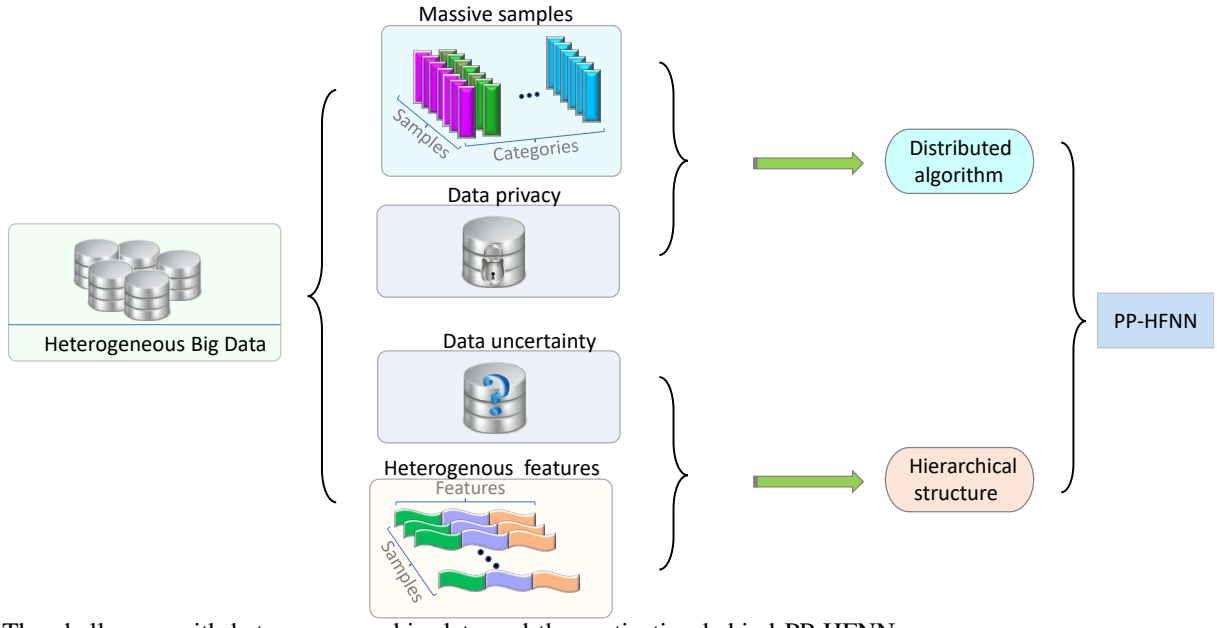


Fig. 1. The challenges with heterogeneous big data and the motivation behind PP-HFNN

processing stage, heterogeneous big data is segmented at both the feature level and the sample level. In the feature-level segmentation, the heterogeneous features are assigned to multiple subsets by a feature splitter to ensure that each subset only contains homogeneous features. Further, each low-level FNN is responsible for only one subset. The samples in the sample-level segmentation for each low-level FNN are further randomly assigned to multiple agents. A distributed computation framework is then applied to these agents in each low-level FNN to preserve data privacy and to deal with the enormous amounts of data samples. The above data processing and PP-HFNN structure is shown in Fig.2.

Suppose $\mathcal{D} := \{(X_i, Y_i)|\ X_i \in \mathbb{R}^{|\mathcal{F}|}, Y_i \in \mathbb{R}, i \in \mathcal{N}\}$ denote the set of heterogeneous data, where $\mathcal{F}$ and $\mathcal{N}$ represents the set of features and samples, respectively, and $|\cdot|$ is the cardinality operator. As shown in Fig.2, the data is first processed by a feature splitter where the full set of heterogeneous features is transformed into multiple independent subsets of homogeneous features and assigned to a low-level FNN. Let $\mathcal{F}_b, b \in \{1, 2, ..., B\}$ denote the subset of homogeneous features associated with the b-th low-level FNN. Then, in each low-level FNN, the training data is further segmented by a sample distributor. $\mathcal{N}_{b,l}$ denotes the subset of training data assigned to the l-th agent ($l \in \{1, 2, ..., L\}$) of the b-th low-level FNN. Accordingly, the sample vector is collected in $X_i^{b,l}$, $\forall i \in \mathcal{N}_{b,l}$ with $x_{i,j}^{b,l}$ denoting its j-th component.

Next, let us briefly recall the structure of the low-level FNNs, which follows the first-order of the Takagi-Sugeno method for fuzzy inference systems and has a layer-by-layer network structure [52]. Now, suppose the output of the l-th agent of the b-th low-level FNN is z_b^l, then the k_b^l-th fuzzy rule can be expressed as

$$\begin{aligned}&\text{Rule } k_b^l\text{: IF } x_{i,1}^{b,l} \text{ is } A_{r,1}^{b,l}, \cdots \text{ and } x_{i,|\mathcal{F}_b|}^{b,l} \text{ is } A_{r,|\mathcal{F}_b|}^{b,l}\\ &\text{Then } z_{r_b^l} = w_{k,0}^{b,l} + \textstyle\sum_{j=1}^{|\mathcal{F}_b|} w_{k,j}^{b,l} x_{j,1}^{b,l},\end{aligned}$$

where $A_{k,j}^{b,l}$ is a Gaussian fuzzy set of the j-th input of rule k_b^l, and $w_{k,j}^{b,l}$ is the corresponding weight of the consequence. The membership function of $A_{k,j}^{b,l}$ can be written as

$$\varphi_{k,j}(x_{i,j}^{b,l}) = \exp\left[-\left(\frac{x_{i,j}^{b,l} - m_{k,j}^{b,l}}{\sigma_{k,j}^{b,l}}\right)^2\right] \tag{1}$$

where $m_{k,j}^{b,l}$ and $\sigma_{k,j}^{b,l}$ are respectively the center and width of the corresponding fuzzy set. Generally, each low-level FNN is composed of four feed-forward layers, as shown on the right-hand side of Fig.2.

Layer 1 is the antecedent layer: its inputs are $x_{i,j}^{b,l}$, and its outputs are derived through the fuzzification process outlined in (1).

Layer 2 is the rule layer: each node in this layer represents a fuzzy rule, which uses the AND operation to match the outputs of the antecedent layer as follows:

$$\phi_k(X_i^{b,l}) = \prod_{j=1}^{|\mathcal{F}_b|} \varphi_{k,j}(x_{i,j}^{b,l}), \tag{2}$$

where $\phi_k(X_i^{b,l})$ is the firing strength of fuzzy rule k_b^l, and then normalized by

$$\bar{\phi}_k(X_i^{b,l}) = \frac{\phi_k(X_i^{b,l})}{\sum_{k=1}^{K_l^b} \phi_r(X_i^{b,l})}. \tag{3}$$

where K_l^b is the total number of fuzzy rules in the l-th agent of the b-th low-level FNN.

Layer 3 is the consequent layer: each node here performs a defuzzification process for each fuzzy rule r_b^l using a weighted average operation as follows:

$$\psi_k(X_i^{b,l}) = \bar{\phi}_k^{b,l}(X_i^{b,l})(w_{k,0}^{b,l} + \sum_{j=1}^{|\mathcal{F}_b|} w_{k,j}^{b,l} x_{j,1}^{b,l}), \tag{4}$$

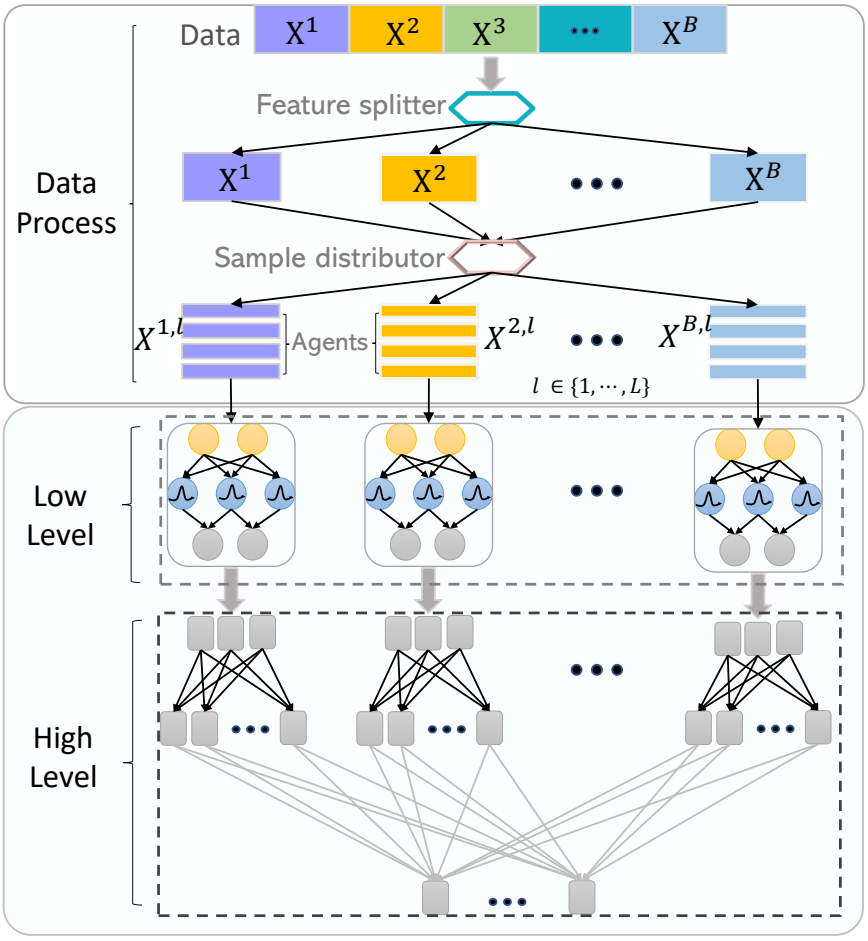


Fig. 2. Data processing and PP-HFNN structure.

Layer 4 is the output layer: the overall output of the l-th agent of the b-th low-level FNN is derived by summing the outputs of the fuzzy rules in Layer 3 as follows:

$$z_i^{b,l} = \sum_{k=1}^{K_l^b} \psi_k^{b,l}(X_i^{b,l}). \tag{5}$$

For each l-th agent of the b-th low-level FNN, a matrix is defined as $H^{b,l} := [H_1^{b,l}, \cdots, H_K^{b,l}]$, where $H_k^{b,l} \in \mathbb{R}^{|\mathcal{N}|\times(|\mathcal{F}_b|+1)}$, and

$$H_k^{b,l} = \begin{bmatrix} \bar{\phi}_k(X_1^{b,l}) & \bar{\phi}_k(X_1^{b,l})x_{1,1}^{b,l} & \cdots & \bar{\phi}_k(X_1^{b,l})x_{1,|\mathcal{F}_b|}^{b,l} \\ \bar{\phi}_k(X_2^{b,l}) & \bar{\phi}_k(X_2^{b,l})x_{2,1}^{b,l} & \cdots & \bar{\phi}_k(X_2^{b,l})x_{2,|\mathcal{F}_b|}^{b,l} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{\phi}_k(X_{|\mathcal{N}|}^{b,l}) & \bar{\phi}_k(X_1^{b,l})x_{|\mathcal{N}|,1}^{b,l} & \cdots & \bar{\phi}_k(X_1^{b,l})x_{|\mathcal{N}|,|\mathcal{F}_b|}^{b,l} \end{bmatrix}$$

The output vector is

$$Z^{b,l} := [z_1^{b,l}, \cdots, z_{|\mathcal{N}|}^{b,l}]^T \in \mathbb{R}^{|\mathcal{N}|}.$$

And the weight vector is

$$\mathbf{w}^{b,l} := [w_{1,0}^{b,l}, \cdots, w_{1,|\mathcal{F}_b|}^{b,l}, \cdots, w_{K_l^b,0}^{b,l}, \cdots, w_{K_l^b,|\mathcal{F}_b|}^{b,l}]^T.$$

Eq. (5) is equivalent to the following matrix form:

$$Z^{b,l} = H^{b,l}\mathbf{w}^{b,l}. \tag{6}$$

It is worth noting that $H^{b,l}$ is dependent on the centers and widths of each rule's fuzzy sets, i.e., $m_k^{b,l}$ and $\sigma_k^{b,l}$ as well as the data input $X_i^{b,l}$.

The outputs of the low-level FNNs are coordinated by a fully-connected layer in the high level of the hierarchy. For each agent l,

$$Y^l = \sum_{b=1}^{B} Z^{b,l} v^{b,l}, \tag{7}$$

where $Y^l := [Y_1^l, \cdots, Y_{\mathcal{N}}^l]$, $v^{b,l}$ is the coordination weight of the hierarchy, and B is the number of subsets in terms of homogeneous features. Let Z^l represent the collection of outputs from each low-level FNN of the l-th agent, i.e.,

$$Z^l := [Z^{1,l}, \cdots, Z^{B,l}] \in \mathbb{R}^{|\mathcal{N}|\times B}.$$

Eq. (8) is equivalent to the following matrix form:

$$Y^l = Z^l\mathbf{v}^l, \tag{8}$$

where

$$\mathbf{v}^l := [v^{1,l}, \cdots, v^{B,l}]^T \in \mathbb{R}^B.$$

Define

$$\mathbf{m}^l := [\mathbf{m}^{1,l}, \cdots, \mathbf{m}^{B,l}],$$
$$\sigma^l := [\sigma^{1,l}, \cdots, \sigma^{B,l}],$$
$$H^l := [H^{1,l}, \cdots, H^{B,l}] \in \mathbb{R}^{|\mathcal{N}|\times n_H},$$

and

$$\mathbf{w}^l := \text{diag}(w^{1,l}, \cdots, w^{B,l}) \in \mathbb{R}^{n_H \times B},$$

where $n_H = \sum_{b=1}^{B} K_l^b(|\mathcal{F}_b| + 1)$, and diag$(\cdot)$ represents a diagonal operation on the matrix. The training procedure for PP-HFNN is to solve the following optimization problem:

$$\min_{\mathbf{m}^l, \sigma^l, \mathbf{w}^l, \mathbf{v}^l} \frac{1}{2} \sum_{l=1}^{L} (||Y^l - H^l \mathbf{w}^l \mathbf{v}^l||^2 + \frac{\lambda}{2}||\mathbf{w}^l||^2 + \frac{\mu}{2}||\mathbf{v}^l||^2), \tag{9}$$

where L is the number of agents, λ and $\mu > 0$ are factors to trade-off the training error and regularization. Selecting an appropriate value for λ and $\mu > 0$ can make the solution much more stable and generalizable [53]. The difficulty of the optimization problem (9) stem from three issues. One is the nonconvex nature of the Gaussian membership function in H^l. To address this first difficulty, the key issue is to identify the parameters of the centers $\mathbf{m}^l$ and the widths σ^l of each rule's fuzzy sets in the antecedent layer of each FNN. The second difficulty is the distributed computing scheme for the low-level FNNs, which is needed to manage the scale of the data and preserve privacy. The last difficulty is the coordination in the high levels of the hierarchy, where the matrix product between $\mathbf{w}^l$ and $\mathbf{v}^l$ is nonconvex. Fortunately, it is convex after fixing either $\mathbf{w}^l$ or $\mathbf{v}^l$, i.e., it becomes bi-convex.

IV. Two-stage optimization algorithm to train the PP-HFNN

An intuitive way to solve the nonconvex optimization (9) is by using a back-propagation method [54]. However, back-propagation methods often suffer from slow training speeds and gradient vanishing problems. In addition, distributed computation is not easily implemented with back-propagation. Hence, it is not an optimal choice for heterogeneous big data environments. A more suitable option is an algorithm that is fast to converge with a massive number of samples and is easy to roll out across a distributed computation framework.

As discussed above, the first difficulty with the optimization problem (9) is to identify the parameters of the antecedent layer of each FNN. A simple idea is to group the input data into multiple clusters and use one rule for each cluster [55]. Here, we use the popular K-means algorithm [56] to identify the parameters of the antecedent layer. The K-means algorithm is one of the most efficient clustering algorithms. To tackle the second difficulty of the optimization problem (9), we developed a distributed K-means method, inspired by [57]. As for the coordination in high level coordination of the hierarchy, the AO method is used since it is well-suited to the bi-convex optimization problems and converges very quickly in practice.

A. Distributed K-means method for the low-level of HFNN

To introduce the distributed K-means algorithm, let us first recall the centralized K-mean algorithm. The goal of this algorithm is to assemble the input data into multiple clusters $\mathcal{C}_k^b$, each of which has its own center. The centralized K-means algorithm for the b-th low-level FNN of the hierarchy can be defined as

$$\mathbf{m}_k^b = \arg\min_{\mathbf{m}_k^b} \frac{1}{2} \sum_{k=1}^{K_b} \sum_{X_i^b \in \mathcal{C}_k^b} ||X_i^b - \mathbf{m}_k^b||^2 \tag{10}$$

where $\mathbf{m}_k^b$ is the k-th center in the antecedent layer of the b-th low-level FNN, and K_b represents the number of corresponding clusters, which is equal to the number of rules. Starting from an initial set of K centers, i.e. $\{\mathbf{m}_1^b(0), \cdots, \mathbf{m}_K^b(0)\}$, the K-means algorithm alternates between an assignment step and an update step as follows.

- Assign each observation X_i^b to the cluster $\mathcal{C}_k^b(t)$, whose center is closest to X_i^b.
- Update the center of each cluster by

$$\mathbf{m}_k^b(t) = \frac{1}{|\mathcal{C}_k^b(t)|} \sum_{X_i^b \in \mathcal{C}_k^b(t)} X_i^b.$$

Once the center of each fuzzy set is obtained by the above procedure, the corresponding standard variance $\sigma_{k,j}^b$ can be calculated as follows:

$$\sigma_{k,j}^b = \sqrt{\frac{1}{|\mathcal{C}_k^b(t)|} \sum_{i=1}^{|\mathcal{C}_k^b(t)|} (x_{i,j}^b - m_{k,j}^b)^2}. \tag{11}$$

It is clear that such a centralized K-means algorithm does not preserve privacy at all, but the distributed K-means algorithm does. As shown in Fig. 3, the data is processed locally by multiple agents. While some limited information is exchanged between agents, none of it is data. The distributed K-means is built on ADMM, which is an efficient solution for distributed computation. A recent study proves the convergence of ADMM for a variety of nonconvex and possibly nonsmooth functions given some sufficient conditions [39]. Following our previous work [51], the distributed K-means algorithm solves the following optimization problem:

$$\min_{\mathbf{m}_K^l} \frac{1}{2} \sum_{l=1}^{L} \sum_{b=1}^{B} \sum_{X_i^b \in \mathcal{C}_k^b} ||X_i^{b,l} - \mathbf{m}_k^{b,l}||^2 \tag{12a}$$

$$\text{s.t.} \quad \mathbf{m}_k^{b,l} = \mathbf{r}_k^b, \quad l = 1, 2, \cdots, L \tag{12b}$$

where $\mathbf{m}_k^{b,l}$ represents the k-th local center of the l-th agent on the b-th low-level FNN, and $\mathbf{r}_k^b$ is the corresponding global center. The global standard variance of the antecedent layer of the low-level FNNs is expressed as

$$\bar{\sigma}_k^b = \sqrt{\frac{1}{|\mathcal{N}|} \sum_{l=1}^{L} |\mathcal{C}_k^{b,l}| (\sigma_k^{b,l})^2} \tag{13}$$

where $\bar{\sigma}_k^b$ denotes the k-th global standard variance of the k-th low-level layer, and $\sigma_k^{b,l}$ is the local standard variance of the l-th agent.

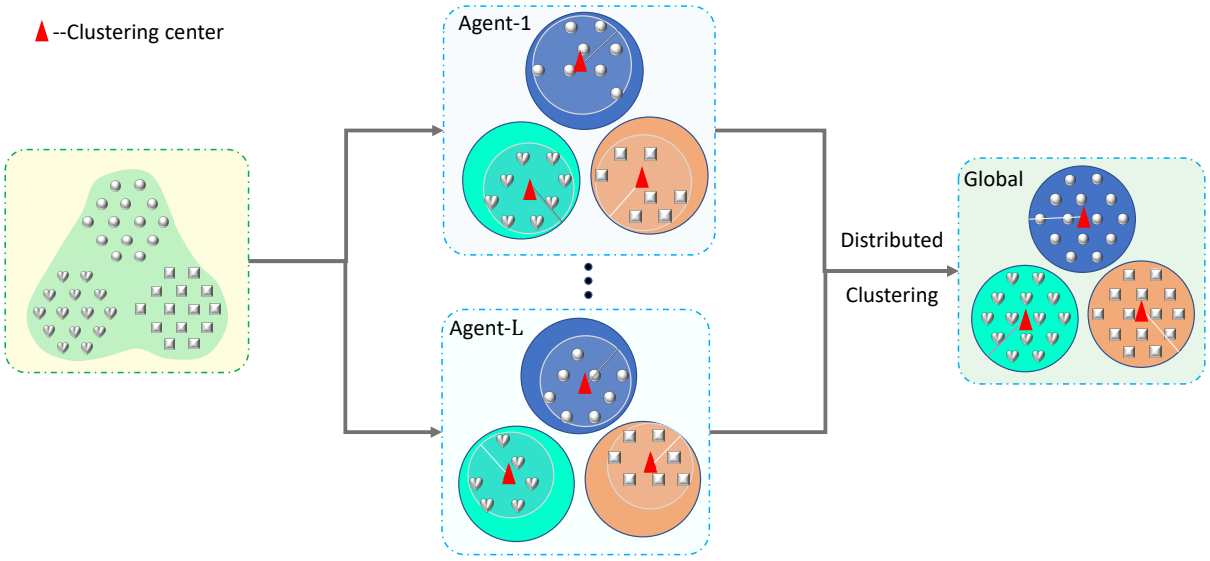


Fig. 3. Distributed Clustering.

The following augmented Lagrangian is contructed for (12):

$$\mathcal{L}(\mathbf{m}_k^{b,l}, \mathbf{r}_k^b, \boldsymbol{\beta}_k^{b,l}) = \frac{1}{2}\sum_{l=1}^{L}\sum_{k=1}^{K}\sum_{X_i^b \in \mathcal{C}_k^{b,l}} ||X_i^{b,l} - \mathbf{m}_k^{b,l}||^2 + \sum_{l=1}^{L}\sum_{k=1}^{K} \boldsymbol{\beta}_k^{b,l^T}(\mathbf{m}_k^{b,l} - \mathbf{r}_k^{b,l}) + \frac{1}{2}\rho\sum_{l=1}^{L}\sum_{k=1}^{K}||\mathbf{m}_k^{b,l} - \mathbf{r}_k^{b,l}||^2 \quad (14)$$

where $\boldsymbol{\beta}_k^l$ is the Lagrange multiplier, and ρ is a positive penalty parameter. The variables are then updated iteratively with the following procedure based on the ADMM:

$$\mathbf{m}_k^{b,l}(t+1) = \arg\min_{\mathbf{m}} \mathcal{L}(\mathbf{m}_k^{b,l}, \mathbf{r}_k^b(t), \boldsymbol{\beta}_k^{b,l}(t)), \quad (15)$$

$$\mathbf{r}_k^b(t+1) = \arg\min_{\mathbf{r}_k^{b,l}} \mathcal{L}(\mathbf{m}_k^{b,l}(t+1), \mathbf{r}_k^b, \boldsymbol{\beta}_k^{b,l}(t)), \quad (16)$$

$$\boldsymbol{\beta}_k^{b,l}(t+1) = \boldsymbol{\beta}_k^{b,l}(t) + \rho\sum_{l=1}^{L}\sum_{k=1}^{K}(\mathbf{m}_k^{b,l}(t+1) - \mathbf{r}_k^b(t+1)), \quad (17)$$

where t is the number of iterations. It is worth noting that the distributed K-means algorithm is applied to each low-level FNN and the centers $\mathbf{m}_k^{b,l}$ can be updated in parallel by different agents. Additionally, there is a closed-form solution for (16) as follows:

$$\mathbf{r}_k^{b,l}(t+1) = \frac{1}{L\rho}\sum_{l=1}^{L}(\boldsymbol{\beta}_k^{b,l}(t) + \rho\mathbf{m}_k^{b,l}(t+1)) \quad (18)$$

Convergence depends on the following two criteria:

$$||\mathbf{m}_k^{b,l}(t) - \mathbf{r}_k^b(t)||^2 \leq \epsilon_1, \quad (19)$$

$$||\boldsymbol{\beta}_k^{b,l}(t+1) - \boldsymbol{\beta}_k^{b,l}(t)||^2 \leq \epsilon_2. \quad (20)$$

B. Alternating optimization for the high-level of HFNN

In the second stage, the AO method is used to obtain the parameters $\mathbf{w}$ and $\mathbf{v}$ in high levels of the hierarchy. Since the raw data is only processed in the first stage, it is not necessary to use distributed computation for the high-level coordination from a privacy-preserving perspective. Therefore, the optimization problem (9) can be simplified to:

$$\min_{\mathbf{w},\mathbf{v}} \frac{1}{2}||Y - H\mathbf{w}\mathbf{v}||^2 + \frac{\lambda}{2}||\mathbf{w}||^2 + \frac{\mu}{2}||\mathbf{v}||^2, \quad (21)$$

which does not incorporate the agents. Obviously, (21) is a bi-convex optimization problem. After fixing either $\mathbf{w}$ or $\mathbf{v}$, it becomes convex. As mentioned, the AO method is a good choice for this bi-convex optimization problem. The process is as follows:

Update $\mathbf{w}$ with fixed $\mathbf{v}(t)$: Let $\hat{\mathbf{w}}$ be the collection of $\{\mathbf{w}^1, \cdots, \mathbf{w}^B\}$ in vector form. By introducing a matrix $H_w(t) := [v^1(t)H^1, \cdots, v^B(t)H^B]$, the weight $\hat{\mathbf{w}}$ can be calculated by solving the following optimization problem:

$$\hat{\mathbf{w}} = \arg\min_{\hat{\mathbf{w}}} \frac{1}{2}||Y - H_w(t)\hat{\mathbf{w}}||^2 + \frac{\lambda}{2}||\hat{\mathbf{w}}||^2, \quad (22)$$

which is a standard least-squares optimization problem. Its closed-form solution can be by setting its partial derivative to 0, i.e.,

$$\hat{\mathbf{w}}(t) = (H_w^T H_w + \lambda I)^{-1} H_w^T Y. \quad (23)$$

Update $\mathbf{v}$ with fix $\mathbf{w}(t)$: The hidden output vector $Z(t)$ of the low-level FNNs is generated after $\mathbf{w}(t)$ is updated. Then the weight $\mathbf{v}(t)$ can be updated by

$$\mathbf{v} = \arg\min_{\mathbf{v}} \frac{1}{2}||Y - Z(t)\mathbf{v}||^2 + \frac{\mu}{2}||\mathbf{v}||^2, \quad (24)$$

Similarly, (24) can be solved through:

$$\mathbf{v}(t) = (Z(t)^T Z(t) + \mu I)^{-1} Z(t)^T Y, \quad (25)$$

A summary of the two-stage PP-HFNN optimization algorithm is provided in Algorithm 1.

Algorithm 1 Two-stage optimization algorithm for PP-HFNN

Stage-1: ADMM-based distributed clustering (12)

Initialization: Set $t = 0$ and randomly initialize the global cluster center $\mathbf{r}_k^b(t)$ and Lagrange multipliers $\boldsymbol{\beta}_k^{b,l}(t)$ for each agent in the low-level layers, and randomly initiate the local cluster centers $\mathbf{m}_k^{b,l}$ for all agents in all low-level layers.

for $t = 0, 1, 2, \cdots$, **do**

Update the local center $\mathbf{m}_k^{b,l}(t+1)$:

Assignment step: Each agent of each low-level layer assigns its data $X_i^{b,l}$ to the cluster $\mathcal{C}_k^{b,l}(t-1)$, derived in the previous iteration.

Update step: Each agent b_l updates the center of each cluster $\mathcal{C}_k^{b,l}(t)$ by

$$\mathbf{m}_k^{b,l}(t+1) = \frac{1}{|\mathcal{C}_k^{b,l}(t)|} \sum_{X_i^{b,l} \in \mathcal{C}_k^{b,l}(t)} X_i^{b,l} \qquad (26)$$

Update the global variables $\mathbf{r}_k^b(t+1)$ by (18) and broadcast it to each agent l.

Update the dual variables $\boldsymbol{\beta}_k^{b,l}(t+1)$ by (17) and broadcast it to each agent l

end for

Stage-2: AO method for high-level layer (9)

Initialization: Set $t = 0$, and randomly initialize the output weight $\mathbf{v}$ and the Lagrange multipliers $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$. Transform $\mathbf{w}$ into $\hat{\mathbf{w}}$

for $t = 0, 1, 2, \cdots$, **do**

Fix $\mathbf{v}$ and update $\hat{\mathbf{w}}$ by (23).

Fix $\hat{\mathbf{w}}$ and update $\mathbf{v}$ by (25).

end for

V. Experimental Evaluation

To evaluate the performance of the proposed PP-HFNN, we first tested it on an artificial dataset and investigated how noise and outliers affected performance. Then, we tested the PP-HFNN on three real datasets with heterogeneous features on both regression and classification tasks. The datasets were: High Storage System Data (HRSS) [58], EEG signals from the Distracted Driving experiment (EEG-DD) [59], and Ethelen-CO (EC) [60]. Since samples of each dataset were either recorded from different types of sensors or assembled from different sources, they are naturally heterogeneous. Details of these datasets are as follows.

The HRSS dataset consists of 6544 records of signal data in two categories captured by 6 sensors equipped to 6 belts of a high storage system. The system generates four datasets according to different belt moving conditions. The features collected by each sensor include transport distance, power, and belt voltage. Therefore, each sample contains 18 features (3 features for each of 6 sensors).

The EEEG-DD dataset contains EEG signals from 11 subjects using 6 channels, which are transformed into 4 bands of frequency data for each channel. Each subject is required to complete two different tasks several times. The signals recorded each time a subject completes one task is called a trial, and each trial is divided into 17 segments. Further, each subject has a different number of trials, ranging from 96 to 183. Thus each segment has 24 features (6 channels divided into 4 bands).

The EC dataset contains time series data of Ethylene and CO readings in the air under dynamic gas mixtures. It contains 4,208,261 samples with 16 features recorded by 16 chemical sensors of 4 different types.

The details of these three datasets are summarized in Table I, where the second, third and fourth column provides the number of features, the low-level FNNs and the samples, respectively.

TABLE I
Dataset and Parameter details

Dataset	Feature	Brunch	Sample	Parameter Setting	
				Agents	Rules
HRSS	18	6	6544	5	1-5
EEG-DD	24	6	-	5	1-15
EC	16	4	4,208,261	200	1-50

To make the simulation more realistic, we conducted both classification and regression tasks. We used the HRSS and EEG-DD datasets for the classification task and the EC dataset with a very large number of samples for the regression task. All classification and regression experiments were conducted on a laptop Intel i7 with a 4.0 GHz processor and 16 GB of memory. As listed in Table I, all the parameters mentioned in the above algorithms were selected by searching the spaces of $[10^{-5}, 10^{-4}, \cdots, 10^{5}]$.

A. Artificial dataset

The artificial dataset we built to investigate the impact of noise and outliers on PP-HFNN's performance was generated as follows. The data input X consisted of two branches of features to stimulate heterogeneous data, i.e., $X = [X^{(1)^T}, X^{(2)^T}]^T$, where $X^{(1)} = [x_1^{(1)}, x_2^{(1)}, x_3^{(1)}]^T$ and $X^{(2)} = [x_1^{(2)}, x_2^{(2)}, x_3^{(2)}]^T$. The corresponding output Y was given by:

$$Y = 0.3\sum_{i=1}^{3}(\cos(x_i^{(1)}))^2 + 0.7\sum_{i=1}^{3}(\cos(x_i^{(2)})), \qquad (27)$$

The feature inputs $X^{(1)}$ were generated from a Gaussian distribution with a mean of $[1, 5, 9]$ and a standard deviation of $[0.1, 0.2, 0.3]$, while $X^{(2)}$ was generated from a Gaussian distribution with a mean of $[3, 7, 2]$ and a standard deviation of $[0.2, 0.4, 0.1]$. We generated a total of 50,000 samples. Three different levels of noise, 5%, 10% and 15%, and three different numbers of outliers, 1%, 5% and 10%, were then added to the feature inputs X. Noise for the feature inputs was generated from a Gaussian distribution $\mathcal{N}(0.1R, (0.1R)^2)$, where $R = max(X) - min(X)$. Similarly, the outliers were generated from a Gaussian distribution $\mathcal{N}(R, (0.1R)^2)$. The data in each low-level branch of PP-HFNN were separated and allocated to five different agents, and 10 fuzzy rules were used. PP-HFNN's performance on the artificial dataset with different levels of noise and outliers is shown in Fig. 4. It is clear that it is quite robust to noise and outliers at various levels.

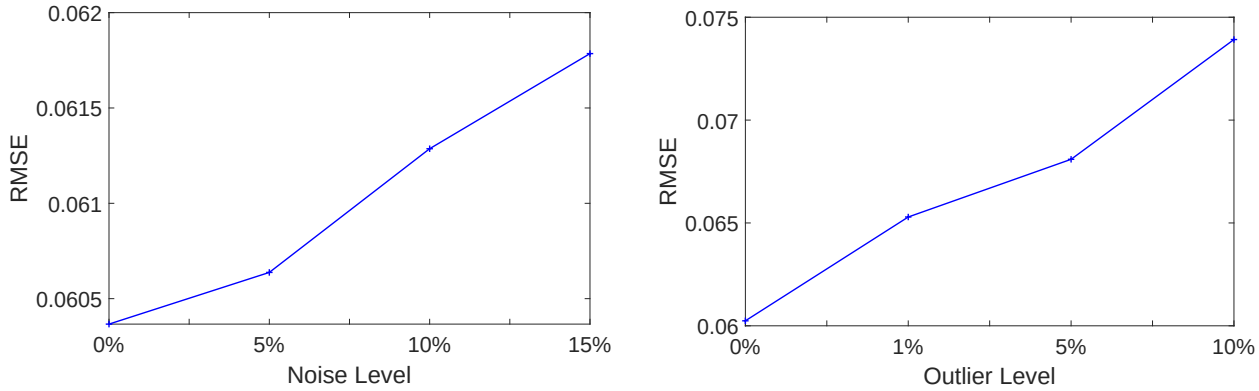


Fig. 4. PP-HFNN's performance on an artificial dataset with different levels of noise (left) and outliers (right)

B. Classification tasks

As mentioned, we evaluated PP-HFNN's classification performance on the HRSS and EEG-DD datasets. Mean average precision (mAP) was the assessment metric, and we compared the results to SVM with different kernel functions. In addition, we also compared PP-HFNN's with a standard non-hierarchical FNN, and an HFNN with a hierarchical structure but not implemented in a distributed scheme. With the HRSS dataset, we randomly segmented the samples into five subsets, each of which was assigned to an agent. Each generated subset was then further split into six subsets according to the feature sources and subjected to five-fold cross-validation.

The features in the EEG-DD dataset relate to the different channels and bands of the signals and, thus, are heterogeneous. However, samples from the same trial are not independent and, therefore, the training data and the test data should not be from the same trial. Also, the EEG-DD dataset is complex with 11 subjects, hundreds of trials, and each trial separated into 17 segments. We adopted the leave-one-out cross-validation method for each subject of the EEG-DD dataset. The mAP value reported is the averaged accuracy of all the subjects. Likewise, with the HRSS dataset, we assigned each subject's data to five agents and report the performance over all subjects in terms of mAP.

TABLE II
SIMULATION RESULTS OF THE CLASSIFICATION TASKS

Dataset	Algorithm	AO	Performance			
			Train(%)	Test(%)	t-test	Time (s)
HRSS	SVM(linear)	-	78.23/0.62	78.10/0.60	0.0001	2.83
	SVM(rbf)	-	90.32/0.42	79.28/0.89	0.0467	3.19
	RF	-	79.08/0.55	78.57/1.21	0.0159	4.17
	FNN	-	79.86/0.54	76.11/0.88	0.0000	9.15
	HFNN	15	**81.15/0.59**	**80.96/1.04**	-	15
	PP-HFNN	15	**80.26/0.57**	**80.30/0.37**	-	18
EEG-DD	SVM(linear)	-	83.80/2.51	76.20/2.41	0.0000	0.24
	SVM(rbf)	-	87.41/7.37	78.06/4.55	0.0012	0.31
	RF	-	87.55/3.26	81.03/2.20	0.0098	1.21
	FNN	-	87.68/2.04	77.44/2.03	0.0000	12.21
	HFNN	20	**87.48/4.34**	**84.94/5.42**	-	16
	PP-HFNN	20	**87.02/4.70**	**84.98/4.02**	-	22

The results for this classification task are provided in Table II. The number of iterations of the AO algorithm is given in the third column. . The training and testing accuracy and corresponding standard variance values are provided in the fourth and fifth columns, respectively. HFNN and PP-HFNN show a better mAP than the other algorithms, which proves that HFNNs can deal with heterogeneous data relatively well. The comparisons between centralized and distributed clustering show that HFNN performed better with a distributed architecture on the EEG-DD dataset, but slightly worse with the HRSS dataset. Overall, however, the results show that a low-level FNN with distributed clustering works better than using a central agent. SVM is a powerful binary classification algorithm but, according to the results in Table II, PP-HFNN has the potential to be better. As Table II shows, PP-HFNN returned a higher mAP than SVM on the EEG DD dataset, which has more heterogeneity than HRSS. From this, we conclude that PP- HFNN is a better option than SVM for highly heterogeneous datasets. In addition, we applied the well-known t-test to evaluate the statistical significance between PP-HFNN and the other methods. The p-values of the t-tests are given in Column 6 of Table II. All p-values were less than 0.05, which indicates that the performance improvements brought by PP-HFNN are statistically significant.

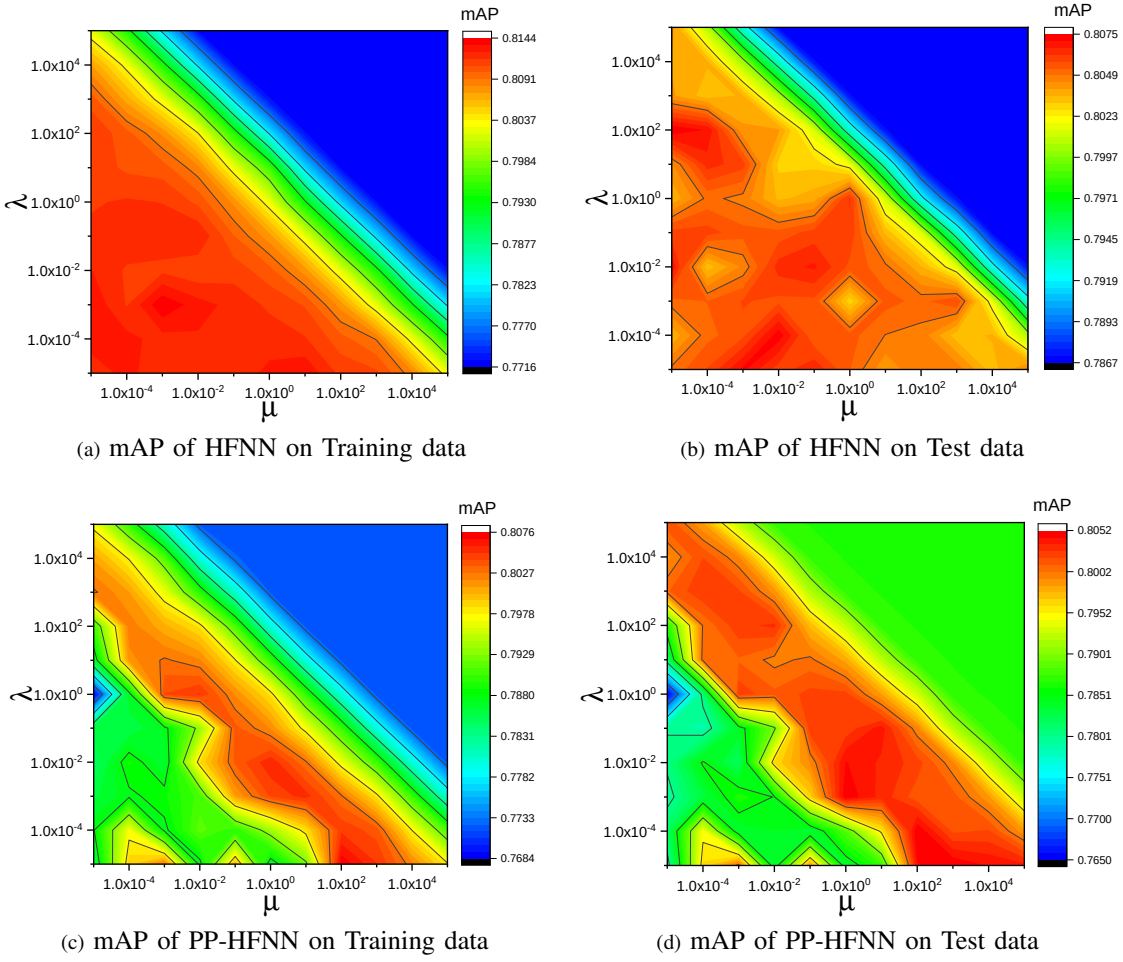
C. Regression tasks

To assess PP-HFNN's performance with regression tasks, we compared it to linear regression, Lasso regression, Ridge regression, polynomial regression and support vector regression (SVR) with the EC dataset and five-fold cross-validation. The degree of polynomial regression ranged from 2 to 5. The evaluation metric was mean square error (MSE), calculated by

$$\text{MSE} = \frac{1}{N\hat{\sigma}_Y}\sum_{i=1}^{N}(\hat{Y}_i - Y_i)^2, \tag{28}$$

The results are listed in Table III.

As shown, HFNN performed better than all the other regression methods, which indicates that a hierarchical structure is more effective for processing heterogeneous data than a non-hierarchical structure. Also, we observed that using a distributed method took almost half the time of the centralized method, which is a significant improvement in time and efficiency. Notably, SVR could not cope with high volumes of data and failed to produce any output. We conducted a t-test for this experiment, too. Again, all p-values were than 0.05



(a) mAP of HFNN on Training data

(b) mAP of HFNN on Test data

(c) mAP of PP-HFNN on Training data

(d) mAP of PP-HFNN on Test data

Fig. 5. Performance of different parameter settings on the HRSS dataset, with each low-level FNN has 4 rules. Results of using HFNN on the (1) training data and (2) test data, and the results of using PP-HFNN on the (1) training data and (2) test data

TABLE III
SIMULATION RESULTS OF THE REGRESSION TASKS

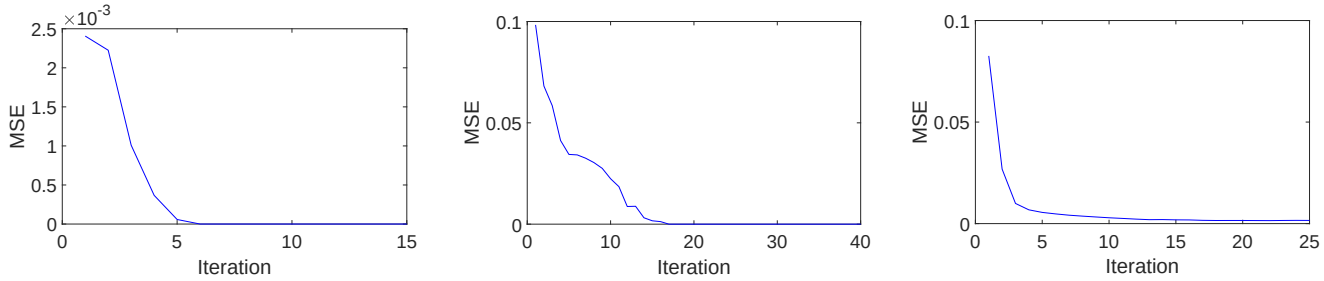
Output	Algorithm	AO	Performance			
			Train	Test	t-test	Time(s)
CO	Linear	-	0.1145/0.0057	0.1139/0.0077	0.0001	11
	Lasso	-	0.1559/0.0050	0.2025/0.0139	0.0000	12
	Ridge	-	0.1176/0.0049	0.1199/0.0174	0.0016	49
	Polynomial	-	0.1034/0.0067	0.1003/0.0075	0.0026	2937
	RF	-	0.0999/0.0043	0.1031/0.0085	0.0018	1851
	HFNN	50	**0.0827/0.0014**	**0.0861/0.0067**	-	3431
	PP-HFNN	50	**0.0770/0.0058**	**0.0806/0.0069**	-	1242
Ethylene	Linear	-	0.1139/0.0061	0.1274/0.0117	0.0000	11
	Lasso	-	0.1673/0.0063	0.1634/0.0143	0.0000	12
	Ridge	-	0.1061/0.0062	0.0974/0.0124	0.0152	49
	Polynomial	-	0.1076/0.0057	0.1035/0.0191	0.0215	2937
	RF	-	0.0999/0.0050	0.0979/0.0204	0.0290	1851
	HFNN	50	**0.0724/0.0036**	**0.0782/0.0058**	-	3431
	PP-HFNN	50	**0.0738/0.0041**	**0.0768/0.0084**	-	1242

(see column 6 Table III), confirming statistical significance of the performance improvements brought by PP-HFNN.

D. Results analysis

It can be seen from Tables II and III that PP-HFNN reached higher accuracy with both the classification and regression tasks than the other tested methods. We believe there are three reasonsfor this: 1) its hierarchical structure, which extracts data features from heterogeneous big data and also reduces the number of fuzzy rules; 2) the distributed K-means algorithm, which is efficient with large-scale data sets and also protects data privacy; and 3) the alternating optimization algorithm, which converges very quickly and does not suffer from the vanishing gradient problem associated with gradient-based learning methods.

We also investigated the trade-off between privacy-preservation and performance by increasing the number of agents from 2 to 10 for the HRSS dataset. The results, presented in Fig. 7, show that the mAP value for PP-HFNN initially increases and then drops as the number of agents increases. In addition, PP-HFNN's performance on the testing data was even better than its centralized counterpart HFNN with less than six agents. Note that the difference between PP-HFNN and HFNN is that PP-HFNN employs a distributed K-means algorithm, while HFNN uses a centralized K-means algorithm to locate the parameters in the antecedent layer. Here, the distributed K-means algorithm can be regarded as a distributed representation learning process, which, in some cases, can outperform its centralized counterpart. However, since the number of training samples allocated to each agent



(a) MSE value of HRSS (b) MSE value of EEG-DD (c) MSE value of Ethylene-CO

Fig. 6. MSE value of distributed K-means using the ADMM.

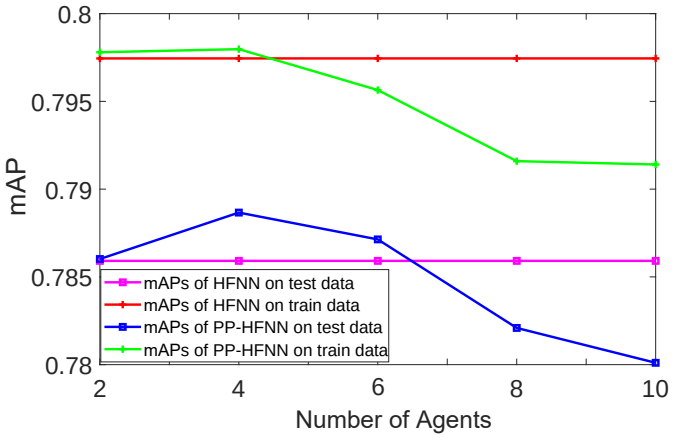


Fig. 7. The performance of PP-HFNN with different number of agents.

decreases as the number of agents increases, in this case, the accuracy of PP-HFNN dropped with greater data privacy protection.

To analyze the influence of the parameter settings, we plotted the mAPs for the HRSS datasets using four rules on each low-level FNN over the parameter space. As shown in Fig. 5, the optimal parameters for PP-HFNN are located in a diagonal area. Additionally, PP-HFNN's mAP stayed above 77% no matter which parameter was used. This is because the low-level FNNs are able to extract informative feature representations, which means the AO method is not sensitive to the parameters. Fig. 6 shows the MSE value curves for the distributed K-means algorithm with all three datasets to analyze convergence. All converged within 20 iterations.

VI. Conclusion

The last few decades have witnessed a big data boom in many areas, such as social networks, the Internet of Things, commerce, astronomy, biology, medicine, etc. However, heterogeneous big data poses a range of challenges to machine learning given its enormous scale and dimensionality and its inherent uncertainty. In this paper, we proposed a privacy-preserving HFNN to address some of the challenges with heterogeneous big data, and especially those surrounding privacy concerns. We presented a two-stage optimization algorithm to train the HFNN. The parameters at low levels of the hierarchy are trained by a distributed K-means algorithm, which does not reveal local data among agents and, thus, is privacy-preserving. Additionally, the distributed K-means algorithm has very good performance, especially with heterogeneous big data at massive scales. Coordination at high levels of the hierarchy is conducted with the AO method, which converges very quickly. The entire training procedure is scalable and does not suffer from slow training speeds or gradient vanishing problems, unlike the methods based on back-propagation. Comprehensive simulations on both regression and classification tasks show that this PP-HFNN outperforms all the tested baseline methods in terms of accuracy and, further, the difference in training time widens as the size of the dataset grows. We are currently exploring a semi-supervised learning method based on the PP-HFNN framework to address scenarios where the training data consists of both labeled and unlabeled samples.

References

[1] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 1, pp. 1–16, 2016.

[2] H.-Y. Tran and J. Hu, "Privacy-preserving big data analytics a comprehensive survey," *Journal of Parallel and Distributed Computing*, vol. 134, pp. 207–218, 2019.

[3] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 19–38, IEEE, 2017.

[4] H. Tuttle, "Facebook scandal raises data privacy concerns," *Risk Management*, vol. 65, no. 5, pp. 6–9, 2018.

[5] D. D. Lewis and J. Catlett, "Heterogeneous uncertainty sampling for supervised learning," in *Machine learning proceedings 1994*, pp. 148–156, Elsevier, 1994.

[6] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[7] L. Wang, "Heterogeneous data and big data analytics," *Automatic Control and Information Sciences*, vol. 3, no. 1, pp. 8–15, 2017.

[8] D. J. Stracuzzi, M. G. Chen, M. C. Darling, S. M. Dauphin, M. G. Peterson, C. Vollmer, and C. J. Young, *Uncertainty in Data Analytics.*, 2016. http://engineering.purdue.edu/~mark/puthesis.

[9] J.-S. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Transactions on systems, man, and cybernetics*, vol. 23, no. 3, pp. 665–685, 1993.

[10] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, and J. Lu, "Fuzzy regression transfer learning in Takagi–Sugeno fuzzy models," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 6, pp. 1795–1807, 2016.

[11] G. Fu and Z. Kapelan, "Fuzzy probabilistic design of water distribution networks," *Water Resources Research*, vol. 47, no. 5, pp. 1–12, 2011.

[12] I. Couso, C. Borgelt, E. Hullermeier, and R. Kruse, "Fuzzy sets in data analysis: From statistical foundations to machine learning," *IEEE Computational Intelligence Magazine*, vol. 14, no. 1, pp. 31–44, 2019.

[13] G. Raju, J. Zhou, and R. A. Kisner, "Hierarchical fuzzy control," *International journal of control*, vol. 54, no. 5, pp. 1201–1216, 1991.

[14] X.-J. Zeng and M. G. Singh, "Approximation theory of fuzzy systems-mimo case," *IEEE Transactions on Fuzzy Systems*, vol. 3, no. 2, pp. 219–235, 1995.

[15] L.-X. Wang, "Analysis and design of hierarchical fuzzy systems," *IEEE Transactions on Fuzzy systems*, vol. 7, no. 5, pp. 617–624, 1999.

[16] R. J. Campello and W. C. do Amaral, "Hierarchical fuzzy relational models: linguistic interpretation and universal approximation," *IEEE Transactions on Fuzzy Systems*, vol. 14, no. 3, pp. 446–453, 2006.

[17] Y. Chen, B. Yang, A. Abraham, and L. Peng, "Automatic design of hierarchical takagi–sugeno type fuzzy systems using evolutionary algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 3, pp. 385–397, 2007.

[18] L.-X. Wang, "Fast training algorithms for deep convolutional fuzzy systems with application to stock index prediction," *IEEE Transactions on Fuzzy Systems*, 2019.

[19] X. Fu, X.-J. Zeng, D. Wang, D. Xu, and L. Yang, "Fuzzy system approaches to negotiation pricing decision support," *Journal of Intelligent & Fuzzy Systems*, vol. 29, no. 2, pp. 685–699, 2015.

[20] P. Brox, I. Baturone, and S. Sánchez-Solano, "Tuning of a hierarchical fuzzy system for video de-interlacing," in *International Conference on Fuzzy Systems*, pp. 1–6, IEEE, 2010.

[21] H. He and J. Lawry, "The linguistic attribute hierarchy and its optimisation for classification," *Soft computing*, vol. 18, no. 10, pp. 1967–1984, 2014.

[22] M. A. ŞAHIN and K. Leblebicioğlu, "A hierarchical fuzzy decision maker for the weapon target assignment," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 8993–8998, 2011.

[23] D. Chen, X. Zhang, L. L. Wang, and Z. Han, "Prediction of cloud resources demand based on hierarchical pythagorean fuzzy deep neural network," *IEEE Transactions on Services Computing*, 2019.

[24] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 439–450, 2000.

[25] K. Chaudhuri and C. Monteleoni, "Privacy-preserving logistic regression," in *Advances in neural information processing systems*, pp. 289–296, 2009.

[26] K. Xu, H. Yue, L. Guo, Y. Guo, and Y. Fang, "Privacy-preserving machine learning algorithms for big data systems," in *2015 IEEE 35th international conference on distributed computing systems*, pp. 318–327, IEEE, 2015.

[27] N. Papernot, P. McDaniel, A. Sinha, and M. P. Wellman, "Sok: Security and privacy in machine learning," in *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 399–414, IEEE, 2018.

[28] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for privacy preserving distributed data mining," *ACM Sigkdd Explorations Newsletter*, vol. 4, no. 2, pp. 28–34, 2002.

[29] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *Journal of Machine Learning Research*, vol. 11, no. May, pp. 1663–1707, 2010.

[30] J. Qin, W. Fu, H. Gao, and W. X. Zheng, "Distributed k-means algorithm and fuzzy c-means algorithm for sensor networks based on multiagent consensus theory," *IEEE transactions on cybernetics*, vol. 47, no. 3, pp. 772–783, 2016.

[31] X. Bi, X. Zhao, G. Wang, P. Zhang, and C. Wang, "Distributed extreme learning machine with kernels based on mapreduce," *Neurocomputing*, vol. 149, pp. 456–463, 2015.

[32] Y. Ye, M. Xiao, and M. Skoglund, "Decentralized multi-task learning based on extreme learning machines," *arXiv preprint arXiv:1904.11366*, pp. 1–11, 2019.

[33] S. Scardapane, D. Wang, and M. Panella, "A decentralized training algorithm for echo state networks in distributed big data applications," *Neural Networks*, vol. 78, pp. 65–74, 2016.

[34] L. Georgopoulos and M. Hasler, "Distributed machine learning in networks by consensus," *Neurocomputing*, vol. 124, pp. 2–12, 2014.

[35] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM Journal on Imaging Sciences*, vol. 1, no. 3, pp. 248–272, 2008.

[36] P. Netrapalli, P. Jain, and S. Sanghavi, "Phase retrieval using alternating minimization," in *Advances in Neural Information Processing Systems*, pp. 2796–2804, 2013.

[37] S. Lu, M. Hong, and Z. Wang, "Pa-gd: On the convergence of perturbed alternating gradient descent to second-order stationary points for structured nonconvex optimization," in *International Conference on Machine Learning*, pp. 4134–4143, 2019.

[38] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[39] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, 2019.

[40] Q. Li, Y. Li, J. Gao, B. Zhao, W. Fan, and J. Han, "Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 1187–1198, 2014.

[41] J. Zhang and G. Huang, "Research on distributed heterogeneous data pca algorithm based on cloud platform," in *AIP Conference Proceedings*, vol. 1967, p. 020016, AIP Publishing LLC, 2018.

[42] D. P. Lewis, T. Jebara, and W. S. Noble, "Support vector machine learning from heterogeneous data: an empirical analysis using protein sequence and structure," *Bioinformatics*, vol. 22, no. 22, pp. 2753–2760, 2006.

[43] Q. Chen, X. Song, H. Yamada, and R. Shibasaki, "Learning deep representation from big and heterogeneous data for traffic accident inference," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[44] H. Zuo, J. Lu, G. Zhang, and W. Pedrycz, "Fuzzy rule-based domain adaptation in homogeneous and heterogeneous spaces," *IEEE Transactions on Fuzzy Systems*, vol. 27, no. 2, pp. 348–361, 2018.

[45] C.-F. Juang, C.-T. Chiou, and C.-L. Lai, "Hierarchical singleton-type recurrent neural fuzzy networks for noisy speech recognition," *IEEE Transactions on Neural Networks*, vol. 18, no. 3, pp. 833–843, 2007.

[46] A. Mohammadzadeh, O. Kaynak, and M. Teshnehlab, "Two-mode indirect adaptive control approach for the synchronization of uncertain chaotic systems by the use of a hierarchical interval type-2 fuzzy neural network," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 5, pp. 1301–1312, 2013.

[47] D. Krleža and K. Fertalj, "Graph matching using hierarchical fuzzy graph neural networks," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 4, pp. 892–904, 2016.

[48] Y. Deng, Z. Ren, Y. Kong, F. Bao, and Q. Dai, "A hierarchical fused fuzzy deep neural network for data classification," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 4, pp. 1006–1012, 2016.

[49] R. Fierimonte, M. Barbato, A. Rosato, and M. Panella, "Distributed learning of random weights fuzzy neural networks," in *2016 IEEE International Conference on Fuzzy Systems*, pp. 2287–2294, IEEE, 2016.

[50] R. Fierimonte, R. Altilio, and M. Panella, "Distributed on-line learning for random-weight fuzzy neural networks," in *2017 IEEE International Conference on Fuzzy Systems*, pp. 1–6, IEEE, 2017.

[51] Y. Shi, C.-T. Lin, Y.-C. Chang, W. Ding, Y. Shi, and X. Yao, "Consensus learning for distributed fuzzy neural network in big data environment," *IEEE Transactions on Emerging Topics in Computational Intelligence*, p. to appear, 2020.

[52] C.-T. Lin and C. S. G. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Transactions on Computers*, no. 12, pp. 1320–1336, 1991.

[53] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, 2011.

[54] C.-F. Juang and C.-T. Lin, "An online self-constructing neural fuzzy inference network and its applications," *IEEE transactions on Fuzzy Systems*, vol. 6, no. 1, pp. 12–32, 1998.

[55] S. L. Chiu, "Fuzzy model identification based on cluster estimation," *Journal of Intelligent & fuzzy systems*, vol. 2, no. 3, pp. 267–278, 1994.

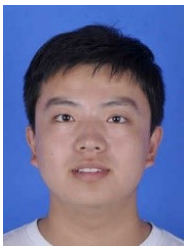
[56] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, Oakland, CA, USA, 1967.

[57] P. A. Forero, A. Cano, and G. B. Giannakis, "Distributed clustering using wireless sensor networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 707–724, 2011.

[58] N. Hranisavljevic, O. Niggemann, and A. Maier, "A novel anomaly detection algorithm for hybrid production systems based on deep learning and timed automata," in *International Workshop on the Principles of Diagnosis (DX)*, 2016.

[59] C.-T. Lin, Y.-K. Wang, and S.-A. Chen, "An eeg-based brain-computer interface for dual task driving detection," in *International Conference on Neural Information Processing*, pp. 701–708, Springer, 2011.

[60] J. Fonollosa, S. Sheik, R. Huerta, and S. Marco, "Reservoir computing compensates slow response of chemosensor arrays exposed to fast varying gas concentrations in continuous monitoring," *Sensors and Actuators B: Chemical*, vol. 215, pp. 618–629, 2015.



Leijie Zhang received the B.S. degree from Hebei Normal University, China, in 2015 and the M.S. degree from Hangzhou Dianzi University, China, in 2019. He is currently pursuing the Ph.D. degree in computer science with the University of Technology Sydney, Ultimo, NSW, Australia. His current research interests include fuzzy neural networks and reinforcement learning.

Ye Shi (M'19) received the B.S. degree in Statistics from Northwestern Polytechnical University, China in 2013 and the Ph.D. degree in electrical engineering at University of Technology Sydney, Australia in 2018. He has been a postdoctoral fellow with the school of computer science at University of Technology Sydney since 2019. His research interests include mixed-integer nonlinear programming, spectral optimization and fuzzy neural networks. He received the Best Paper Award at the 6th IEEE International Conference on Control Systems, Computing and Engineering in 2016.

Yu-Cheng Chang received the B.S. degree in vehicle engineering from the National Taipei University of Technology, Taipei, Taiwan, in 2008, the M.S. degree with a specialization in system and control from the Department of Electrical Engineering, National Chung-Hsing University (NCHU), Taichung, Taiwan, in 2010. From 2014 to 2016, He had been a Research Assistant with the Department of Electrical Engineering, NCHU. He is perusing his Ph.D. degrees in computer science at University of Technology Sydney since 2016. His current research interests include fuzzy systems, human-machine autonomous, and brain state analysis.

Chin-Teng Lin (F'05) received the B.S. degree from National Chiao-Tung University (NCTU), Taiwan in 1986, and the Master and Ph.D. degree in electrical engineering from Purdue University, USA in 1989 and 1992, respectively. He is currently the Distinguished Professor of Faculty of Engineering and Information Technology, and Co-Director of Center for Artificial Intelligence, University of Technology Sydney, Australia. He is also invited as Honorary Chair Professor of Electrical and Computer Engineering, NCTU, and Honorary Professorship of University of Nottingham. Dr. Lin was elevated to be an IEEE Fellow for his contributions to biologically inspired information systems in 2005, and was elevated International Fuzzy Systems Association (IFSA) Fellow in 2012. Dr. Lin received the IEEE Fuzzy Systems Pioneer Awards in 2017. He served as the Editor-in-chief of IEEE Transactions on Fuzzy Systems from 2011 to 2016. He also served on the Board of Governors at IEEE Circuits and Systems (CAS) Society in 2005-2008, IEEE Systems, Man, Cybernetics (SMC) Society in 2003-2005, IEEE Computational Intelligence Society in 2008-2010, and Chair of IEEE Taipei Section in 2009-2010. Dr. Lin was the Distinguished Lecturer of IEEE CAS Society from 2003 to 2005 and CIS Society from 2015-2017. He serves as the Chair of IEEE CIS Distinguished Lecturer Program Committee in 2018 2019. He served as the Deputy Editor-in-Chief of IEEE Transactions on Circuits and Systems-II in 2006-2008. Dr. Lin was the Program Chair of IEEE International Conference on Systems, Man, and Cybernetics in 2005 and General Chair of 2011 IEEE International Conference on Fuzzy Systems. Dr. Lin is the coauthor of Neural Fuzzy Systems (Prentice-Hall), and the author of Neural Fuzzy Control Systems with Structure and Parameter Learning (World Scientific). He has published more than 330 journal papers (Total Citation: 22,913, H-index: 68, i10-index: 290) in the areas of neural networks, fuzzy systems, brain computer interface, multimedia information processing, and cognitive neuro-engineering, including about 125 IEEE journal papers.