Tensor Deep Learning Model for Heterogeneous Data Fusion in Internet of Things

Wei Wang*, Min Zhang

Abstract*—With the rapid evolvement of the Internet and data acquisition technology as well as the continuous advancement of science and technology, the amount of data in many fields has reached the level of terabyte or petabyte and most data collection comes from the Internet of Things (IoT). The rapid advancement of IoT big data has provided valuable opportunities for the development of people in all areas of society. At the same time, it has also brought severe challenges to various types of current information processing systems. Effectively using the big data technology, discovering the hidden laws in big data, tapping the potential value of big data, and predicting the development trend of things to allocate resources more reasonably will promote the overall development of society. However, most of the IoT big data are presented as heterogeneous data, with high dimensions, different forms of expression, and a lot of redundant information. The current machine learning model works in vector space, which makes it impossible to gain big data features because vectors cannot simulate the highly nonlinear distribution of IoT big data. This article presents a deep learning calculation model called tensor deep learning (TDL), which further improves big data feature learning and high-level feature fusion. It uses tensors to model the complexity of multi-source heterogeneous data and extends the vector space data to the tensor space, when feature extraction in the tensor space is included. To fully understand the underlying data distribution, the tensor distance is adopted as the average square sum error term of the output layer reconstruction error. Based on the conventional back-propagation algorithm, this study proposes a high-order back-propagation algorithm to extend the data from the linear space to multiple linear space and train the parameters of the proposed model. Then, to evaluate its performance, the proposed TDL model is compared with the stacked auto encoder and multimodal deep learning model. Furthermore, experiments are performed on two representative datasets, namely CUAVE and STL-10. Experimental results show that the proposed model not only excels in heterogeneous data fusion but also provides a higher recognition accuracy than the conventional deep learning model or the multimodal learning model for big data.

***Index Terms*—Big Data, Heterogeneous Data Fusion, Tensor Feature Extraction, Tensor Deep Learning Model.**

I. Introduction

With the rapid evolvement of the network technologies and Internet information, the Internet of Things (IoT) came into being and attracted the attention of researchers in related fields. The IoT is an information-sensing device that collects required information in real time through infrared sensors, radio frequency identification, global positioning systems, and laser scanners. It provides people with transparent, smart, ubiquitous, green, and secure services and is widely used in many applications such as anti-intrusion systems, food safety control, modern logistics management, positioning navigation, and many other applications [1]. However, these IoT big data representations are diverse, such as images, text, numbers, audio, and video. Most of these are structured data, semi-structured data, and unstructured data, which are highly heterogeneous in terms of structure [2]. In the real world, observation goals for the same semantic concept ontology can often adopt multiple observation methods to obtain data information from various observation channels. These data from different information channels describe the same concept ontology. Each type of information in the data or each observation angle can be called the sample information. Different types of sample information form the heterogeneous data for the same problem [3]. Data fusion is essentially the collaborative processing of multi-source data for achieving redundancy reduction, comprehensive complementarity, and collaborative information capturing. This technology has attracted wide attention in the field of data processing, Target recognition, Disaster relief, and intelligent decision-making [4]. When describing the concept ontology, the use of multiple data features can often be more comprehensive and detailed. Therefore, how to use multiple heterogeneous data information simultaneously to extract effective heterogeneous data fusion features from multiple sample information needs to be investigated. The task of information fusion and pattern recognition is extremely important and is a problem that requires further study.

Large-scale mass data with the feature extraction, people expect to obtain intrinsic and implicit information from data, and it is not limited to the study of surface relationships. Some Conventional data reduction methods, such as Principal Component Analysis [5,6] and Linear Discriminant Analysis [7] are used to transform the original data in space, and the dimension of the original data is extracted from the low dimension data from the high dimension, and extract new low-

This paper is supported by Natural Youth Science Foundation of China (61501326,61401310), the National Natural Science Foundation of China (61731006) and Natural Science Foundation of China (61271411). It also supported by Tianjin Research Program of Application Foundation and Advanced Technology (15JCZDJC31500), and Tianjin Science Foundation(16JCYBJC16500). This work was supported by the Tianjin Higher Education Creative Team Funds Program.

The authors are with the Laboratory of Wireless Communication and Energy Transmission, Tianjin Normal University, 300387, CHINA. (e-mail: weiwang@tjnu.edu.cn).

dimensional features to achieve dimension reduction. However, this method will destroy the hidden internal links between the original data, leading to a negative effect on data pattern recognition.

For multi-source heterogeneous data fusion, Nakamura, Loureiro, and Frery [8] used the conventional fusion method based on statistics and artificial intelligence methods to study the multi-sensor data fusion technology. Li, Zhang, and Pan [9] constructed a multi-source nonlinear heterogeneous data fusion model for studying multi-source heterogeneous data in mobile physics information systems, organization, and management. In [10], ontology was employed to extract each node and attribute of RDF(S) ontology concerning multiple source heterogeneous data of EMU for the whole life cycle to perform inference calculations, thereby obtaining converged heterogeneous data. In [11], tensor factorization was adopted to extract the core features of neuroimaging obtained by ROI or body-based methods for Alzheimer's disease (AD); multinuclear learning methods are combined to detect AD disease to achieve early detection and help patients effectively avoid or reduce its incidence to improve people's living standards. Ling [12] combined wireless sensor networks and data fusion technology, and proposed a Kalman filter batch estimation fusion algorithm, which was successfully applied to the process of target location tracking. In [13], the wavelet-moment feature and the back-propagation (BP) neural network were adopted to fuse airborne radar and infrared sensors, and the maximum membership degree rule was used for decision-making and classification to achieve better recognition results. Li, Hu, and Chen [14] studied the fusion technology for multi-source heterogeneous data in the digital mine construction process, which ensures the security, stability, and efficiency of the basic information platform in digital mine construction. Hu et al. [15] studied the massive multimodal data fusion method in the IoT network environment, and it was used successfully in the target location tracking process. The conventional fusion method has been relatively mature, but with the boom of human–robot wars and Alpha Dogs, deep learning has attracted much attention. Multi-source heterogeneous data fusion is much higher than traditional data fusion methods in terms of redundant data processing, recognition accuracy and classification accuracy. A typical deep learning model consists of the following three basic models: deep belief network, automatic encoder and convolutional neural network. Other models can be considered as variants of the three basic models mentioned above. Ngiam et al. [16] introduced deep learning into feature extraction of multimodal data, integrated two different information modalities of audio and video on speech recognition data, and trained a deep belief network to extract the joint feature from two modalities; they achieved good results in the task of video semantic understanding. Srivastava and Salakhutdinov [17] applied the deep neural network to an image retrieval task, and various features extracted from the image data were used as the image modalities. The image text annotations made by users were employed as text modalities to construct the neural network model of the same depth. Using these two modalities for training and at the same time for obtaining a joint high-level abstract semantic feature for image classification and retrieval has also achieved good results. However, the conventional deep learning methods are also adopted to learn core features from each mode and integrate different levels of learning features into a shared representation of multimodal data. The multimodal deep learning (MDL) model can effectively capture the high-order correlations between multiple modes, thereby forming a hierarchical representation of multimodal data. However, it cannot simulate the nonlinear distribution of the input heterogeneous data, because the features are learned from different modal data independently, resulting in the inability to learn the useful features of big data, making heterogeneous data fusion accuracy at the feature level is not high.

In order to settle these problems, in this article, the tensor deep learning (TDL) model, which is a depth calculation model for performing feature learning and forming deep levels of representations of big data-based tensors space is proposed. First, a tensor-based data representation model is adopted to connect and represent various heterogeneous data in a unified way. This tensor-based representation method can simulate the high-order correlation of big data [18]. Based on such a tensor-based representation model, a tensor auto-encoder (TAE) is proposed as the base module for the TDL model. TAE uses the tensor distance instead of the Euclidean metric and cross-entropy as the reconstructed error mean and squared error terms to stimulate intermediate expressions and further capture the unknown high-dimensional distribution of big data as much as possible. In addition, a high-order back-propagation (HBP) algorithm as an extension of the conventional BP algorithm in the high-order tensor space is designed for training the parameters in TAE [19]. Finally, a TDL model is built by stacking multiple TAEs to learn multiple levels of features on big data and fuse heterogeneous data of multiple modalities. It can be clearly seen from the experimental results that the recognition accuracy is high. It is clearer that the main ideas of heterogeneous data fusion in this paper are as follows:

1) Extraction of features by tensor factorization. This study extends the data in the linear space to multiple linear space. Tensors can maintain the internal structure of data and have been widely used in machine learning, recommendation system, image processing, Pattern recognition, signal processing, and other fields. The high-order singular value decomposition method (HO-SVD) can be adopted to extract core data features from low-quality and complex raw data [20]. As the HO-SVD method has orthogonality requirements for truncated vector bases, it can be considered as a special form of Tucker decomposition. The core tensor obtained by the HO-SVD method and the truncated orthogonal basis of each modular expansion matrix can be regarded as the core dataset [21]. The reconstructed approximation tensor contains better data to achieve the data feature extraction.

2) Heterogeneous data fusion through the TDL model. In this study, the features derived from the tensor factor decomposition are used as the input data of the TDL model, and heterogeneous data feature fusion is performed on high-level semantic features of the data. The TDL model is extended from the linear space to multiple linear space, and a TAE model based on tensor data

identification is constructed to solve the parameters of the high-order auto-encoder model. Finally, multiple high-order auto-encoder models are stacked to build a TDL model for heterogeneous big data feature learning [22].

As far as we know, tensor factorization is used for extracting features from multi-source heterogeneous data, removing redundancy, and obtaining core features. The TDL model is adopted for integrating heterogeneous data of multiple modalities at the feature level.

The rest of the paper is organized as follows. In Section 2, the tensor basic theoretical knowledge and feature extraction in high-dimensional space are introduced, and a TDL model is proposed for the heterogeneous data fusion in Section 3. The fourth section gives the results and analysis of the experiment. Finally, Section 5 concludes the discussion and specifies future research related to this study.

II. Feature Extraction through Tensor decomposition

Currently, in the calculation process of multi-source heterogeneous big data, the existence of large amounts of inconsistency, repeated redundancy, and noise data seriously affects the efficiency of large-data processing algorithms and accuracy of calculation results. Constructing a concise mathematical model to represent complex data in a unified manner and designing efficient and secure feature extraction algorithms to extract high-quality core datasets from low-quality raw big data have great theoretical and practical significance for big data research. The conventional feature extraction algorithm is limited to the study of surface relationships and cannot obtain the internal connection of heterogeneous data. An effective way of overcoming these problems is to use a set of multi-source heterogeneous data as a tensor and apply it to tensor factorization. We present below the advantages of tensor-based data representation compared with vector data representation [23]:

(1) With the tensor structure, it is intuitively easy to represent multi-source heterogeneous data, where first-order represents a modality, and there are different modal dimensions. Vector representation often overlooks this point.

(2) When extracting features (especially image features), the tensor can effectively preserve the structural features of the original sample.

(3) In data transformation, the tensor can fully consider the correlation and complementarity among different modalities, but it is difficult to determine the above relationship by vector representation.

(4) The tensor can effectively solve the difficult problems of vector representation, such as dimension disasters and matrix singularity.

There are two methods of extracting features from a tensor space: 1) the tensor space is projected into a tensor subspace, and the tensor is transformed into one with the same order but a lower dimension. Then, the vectorization of the tensor at a low latitude is achieved by converting it into an eigenvector. This method is mainly implemented by the tensor decomposition method, such as the tensor Tucker decomposition method [24]; 2) the tensor space is projected into a vector space, and the tensor is directly transformed into an eigenvector. This method is mainly implemented by tensor discriminant analysis, such as rank 1 tensor discriminant analysis [25]. In this study, the first method is chosen.

A. *Basic Information on Tensors*

As a high-dimensional array, tensors are widely used in data mining, machine learning, face recognition, and other fields. In mathematics, the tensor is expressed as a high-order expansion of a vector in the spatial dimension. For example, in the isomorphic sense, the zero-order tensor is called the scalar, the vector is the first-order tensor, and the matrix is called the second-order tensor [26]. We provide the general form of tensors, i.e., higher-order tensors. The K th-order tensor is expressed as $T \in R^{I_1 \times I_2 \times \cdots \times I_k}$, and the tensor T element is represented as $t_{i_1 \cdots i_n \cdots i_k}$, where $1 \le i_k \le I_k,\ for\ \mathrm{n} = 1, \cdots, \mathrm{K}$. If a K th-order tensor T can be expressed as an outer product of T vectors, then the vector is called a rank quantity.

$$T = x_1 \otimes x_2 \otimes \cdots \otimes x_K \tag{1}$$

Here, $x_k \in R^{I_k}$, $for\ all\ 1 \le n \le K$.

The tensor and matrix single-mode multiplication is the generalization of two matrix products in the vector space. The elements of the tensor $T \in R^{I_1 \times I_2 \times \cdots \times I_K}$ and matrix $P \in R^{J_k \times I_k}$ k -mode products are expressed as $P_{j_k i_k}$, where $1 \le j_k \le J_k$ and $1 \le i_k \le I_k$ are tensors denoted as

$$T \times_k P \in R^{I_1 \times \cdots \times I_{k-1} \times J_k \times I_{k+1} \times \cdots \times I_K} \tag{2}$$

whose entries are given by

$$\left(T \times_k P\right)_{i_1 \cdots i_{k-1} j_k i_{k+1} \cdots i_K} = \sum_{i_k} t_{i_1 \cdots i_{k-1} i_k i_{k+1} \cdots i_K} p_{j_k i_k} \tag{3}$$

In terms of multi-module multiplication operation of tensors, let $Q \in R^{I_1 \times I_2 \times \cdots I_K}$ be another tensor, and its general element is represented as $q_{i_1 \cdots i_k \cdots i_K}$. Tensor T is multiplied by tensor Q along some specified orders to obtain a new tensor, denoted as

$$\langle T, Q \rangle = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_K} t_{i_1 \cdots i_k \cdots i_K} q_{i_1 \cdots i_k \cdots i_K} \tag{4}$$

The Frobenius norm of a tensor is defined as

$$\|T\| = \sqrt{\langle T, T \rangle} \tag{5}$$

The feature extraction and feature reduction of the data represented in the tensor space are obtained by reduction in the data tensor rank. The tensor rank is defined based on the tensor decomposition. The tensor decomposition includes the CP decomposition and Tucker decomposition. In the next section, we explain in detail the Tucker decomposition. The k -th grade of T denoted as $rank_k(T)$ is defined as the dimension of the vector space spanned by the k pattern vectors: $rank_k(T) = rank(T_{(k)})$.

B. Tensor Factorization

Given a tensor $T \in R^{I_1 \times I_2 \times \cdots \times I_K}$, the $rank-(R_1,\cdots,R_K)$ factorization of T is formulated as finding a lower-rank tensor. Next, we introduce the core tensor (S) and the approximate tensor ($\hat{T}$).

The approximate tensor $\hat{T}$ can be expressed as follows:

$$\hat{T} = S \times_1 U^{(1)} \times_2 U^{(2)} \times \cdots \times_K U^{(K)} \tag{6}$$

The matrix $U^{(k)} \in R^{I_k \times R_k}$, where $k = 1,\cdots,K$, corresponds to the truncated left singular vector matrix obtained by singular value decomposition (SVD) of the tensor matrices. These matrices are also called truncated orthogonal base spaces of the tensor T. The core tensor S and truncated orthogonal base space $\left\{U^{(k)}\right\}_{k=1}^{K}$ form the core tensor of the original tensor T, when R_k is much smaller than I_k, resulting in data compression.

The core tensor S can be expressed as follows:

$$S = T \times_1 (U^{(1)})^T \times_2 (U^{(2)})^T \cdots \times_K (U^{(K)})^T \tag{7}$$

It can be observed from equations (6)–(7) that given the base matrices $\left\{U^{(k)}\right\}_{k=1}^{K}$, the core tensor S can be calculated very quickly. Therefore, the optimization problem only focuses on the calculation of the truncated orthogonal base space $\left\{U^{(k)}\right\}_{k=1}^{K}$.

In [27,28] iterative methods are applied to tensor decomposition. Only one of the truncated orthogonal base spaces is optimized in each iteration while the other $K-1$ truncated orthogonal base space remains fixed.

The above iterative algorithm solution depends on the initialization and is computationally expensive. In this study, we apply the SVD of the tensor and a generalization of matrix SVD, also known as HO-SVD [25]. This algorithm has been successfully applied to image processing and computer vision [20]. First, define D_k as an $I_k \times I_k$ matrix, whose (u,v) -th entry ($1 \le u,v \le I_k$) is given by

$$\sum_{i_1}\cdots\sum_{i_{k-1}}\sum_{i_{k+1}}\cdots\sum_{i_K} t_{i_1\cdots i_{k-1}ui_{k+1}\cdots i_K}\cdot t_{i_1\cdots i_{k-1}vi_{k+1}\cdots i_K} \tag{8}$$

where the Kth-order tensor $T \in R^{I_1 \times I_2 \times \cdots \times I_K}$ elements are $t_{i_1\cdots i_{k-1}ui_{k+1}\cdots i_K}$ and $t_{i_1\cdots i_{k-1}vi_{k+1}\cdots i_K}$. It follows that D_k is a symmetric and positive semi-definite matrix. Let $D_k = U_k \Sigma_k U_k^T$ be the SVD of D_k. Then, denote $U^{(k)}$ as the truncated orthogonal base space, which consists of the first R_k columns of U_k.

When using SVD for Tucker1 decomposition, HO-SVD cannot guarantee a good approximation, but we can use the result of HO-SVD as an initial solution of higher-order orthogonal iteration to iterate through reconstruction errors and obtain better results [29]. Fig. 1 shows a HO-SVD scheme for a third-order tensor decomposition model.

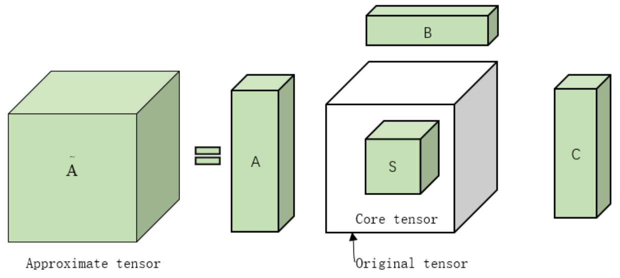

Fig.1 Third-Order Tensor factorization

C. Tensor Distance

The tensor distance is used to measure the distance between two tensors in the high-order tensor space, which can reveal the real difference between two data objects for high-order, complex data by modeling the correlation between different data coordinates with any number of orders [30].

Given a K -order tensor $X \in R^{I_1 \times I_2 \times \cdots \times I_K}$ and $Y \in R^{I_1 \times I_2 \times \cdots \times I_K}$, x is a vector expression of X, and the element $X_{i_1 i_2 \cdots i_K (1 \le i_j \le I_j, 1 \le j \le K)}$ in X corresponds to x_l, i.e., the l -th element in x, where $l = i_1 + \sum_{j=2}^{K} \prod_{t=1}^{j-1} I_t$. Tensor Y is similar to tensor X, and then, the tensor distance between two K -order tensors is defined as [31]

$$d_{TD} = \sqrt{\sum_{l,m=1}^{I_1 \times I_2 \times \cdots \times I_K} g_{lm}(x_l - y_l)(x_m - y_m)} = \sqrt{(x-y)^T G (x-y)} \tag{9}$$

where g_{lm} is a coefficient and G is the coefficient matrix, which reflects the internal relationship among different coordinates of the high-order data and is defined as follows

$$g_{lm} = \frac{1}{2\pi\sigma^2} \exp\left\{-\frac{\left\|p_l - p_m\right\|_2^2}{2\sigma^2}\right\} \tag{10}$$

Where, $\left\|p_l - p_m\right\|_2$ is the location distance between $X_{i_1 i_2 \cdots i_K}$ (corresponding to x_l) and $X_{i_1' i_2' \cdots i_K'}$ (corresponding to x_m), defined as

$$\left\|p_1 - p_m\right\|_2 = \sqrt{(i_1 - i_1')^2 + (i_2 - i_2')^2 + \cdots + (i_N - i_K')^2} \tag{11}$$

III. Multi-source Heterogeneous Data Fusion Model

This study uses the tensor factor decomposition to extract high-level semantic features of data. It is applied to the depth calculation model based on tensor data: the TDL model is a tensor-based model for depth calculation of big data feature learning. As the multi-source heterogeneous data describing the same ontology have some correlation in the high-level semantic features and the data are extended from the linear space to multiple linear space, the data internal structure including the integrity of data features can be preserved. The tensor depth calculation model consists of a stack of multiple tensor automatic encoders. Next, we first describe the basic principles of the TAE.

A. Tensor Auto-encoder Model

The TAE is similar to the basic automatic encoder [32] and is featured with one input layer, multiple hidden layers, and one output layer. Unlike the basic automatic encoder, the TAE is represented by tensors for each layer of data. According to the basic principle of neural network, the data obtained in the previous layer, and then the output result is passed to the next layer through the sigmoid function, and finally the output of each hidden layer and the output of the output unit are determined.

In Section 2, the basic definitions of tensors and related theories are given; however, in order to formalize the description of higher-order auto-encoder models, first, the multi-point products of the tensor should be reviewed, which could be conducive to the clear description of the TAE model.

Definition 1. Multi-dot product ($\odot$). Given an $N+1$ -order tensor $A \in R^{\alpha \times I_1 \times I_2 \times \cdots I_N}$ and an N order tensor $B \in R^{I_1 \times I_2 \times \cdots I_N}$, where A has α sub-tensor, each of which is denoted by $A_\beta \in R^{I_1 \times I_2 \times \cdots I_N}$. The multi-point product of A and B is an N -order tensor $H \in R^{J_1 \times J_2 \times \cdots \times J_N}$ $\left(J_1 \times J_2 \times \cdots \times J_N = \alpha\right)$, and $H = A \odot B$. Each element in H is defined as follows

$$h_{j_1 j_2 \cdots j_n} = A_\beta \cdot B(\beta = j_n + \sum_{i=1}^{N-1} (j_i - 1) \prod_{t=i+1}^{N} J_t) \quad (12)$$

Now we present a formal description of the TAE model. Two N -order tensors $X \in R^{I_1 \times I_2 \times \cdots I_N}$ and $H \in R^{J_1 \times J_2 \times \cdots J_N}$ represent the input layer data and hidden layer data respectively. According to the principle of auto-encoder, the input layer X after the activation function f maps the hidden layer H :

$$H = f_\theta(W^{(1)} \odot X + b^{(1)}) \quad (13)$$

where $\theta = (W^{(1)}, b^{(1)})$ is the encoder parameter set; $W^{(1)} \in R^{\alpha \times I_1 \times I_2 \times \cdots \times I_N}$ is an $N+1$ -order tensor with α sub-tensors $\left(\alpha = J_1 \times J_2 \times \cdots \times J_N\right)$, each of which is denoted by $W_\mu^{(1)} \in R^{I_1 \times I_2 \times \cdots \times I_N}$; $b^{(1)} \in R^{J_1 \times J_2 \times \cdots \times J_N}$ is an N -order tensor, and f is the activation function (sigmoid function), i.e., $f\left(x\right) = 1/\left(1 + e^{-x}\right)$ [33].

The decoder function g maps the hidden layer data back to reconstruction Y :

$$Y = h_{W,b}\left(X\right) = g_\theta\left(W^{(2)} \odot \mathrm{H} + b^{(2)}\right) \quad (14)$$

where $\theta = (W^{(2)}, b^{(2)})$ is the decoder parameter set; $W^{(2)} \in R^{\beta \times J_1 \times J_2 \times \cdots \times J_N}$ is an $N+1$ -order tensor with β sub-tensors $\left(\beta = I_1 \times I_2 \times \cdots \times I_N\right)$, each of which is denoted by $W_\mu^{(2)} \in R^{J_1 \times J_2 \times \cdots \times J_N}$, when $b^{(2)} \in R^{I_1 \times I_2 \times \cdots \times I_N}$ is an N -order tensor, and g is also a sigmoid function [33].

In order to train the parameters of the high-order automatic encoder, the BP algorithm needs to be extended from the linear space to multiple linear space. In order to derive the solution of the BP algorithm [34] in the tensor space, the representation of the high-order automatic coder model is given.

Given the input tensor $X \in R^{I_1 \times I_2 \times \cdots I_N}$, $z_{j_1 j_2 \cdots j_N}^{(2)}$ $\left(1 \le j_i \le J_i, 1 \le i \le N\right)$ indicates the hidden layer input data; $z_{i_1 i_2 \cdots i_N}^{(3)}$ $\left(1 \le i_j \le I_j, 1 \le j \le N\right)$ indicates the output layer input data; $a_{j_1 j_2 \cdots j_N}^{(2)}$ $\left(1 \le j_i \le J_i, 1 \le i \le N\right)$ indicates the hidden layer data, and $a_{i_1 i_2 \cdots i_N}^{(3)}$ $\left(1 \le i_j \le I_j, 1 \le j \le N\right)$ indicates the output layer data. Their relationship is as follows:

$$z_{j_1 j_2 \cdots j_n}^{(2)} = W_\beta^{(1)} \odot X + b_{j_1 j_2 \cdots j_n}^{(1)} (\beta = j_n + \sum_{i=1}^{N-1} (j_i - 1) \prod_{t=i+1}^{N} J_t) \quad (15)$$

$$a_{j_1 j_2 \cdots j_n}^{(2)} = f\left(z_{j_1 j_2 \cdots j_n}^{(2)}\right) \quad (16)$$

$$z_{i_1 i_2 \cdots i_n}^{(3)} = W_\alpha^{(2)} \odot a^{(2)} + b_{i_1 i_2 \cdots i_n}^{(2)} (\alpha = i_n + \sum_{j=1}^{N-1} (i_j - 1) \prod_{t=j+1}^{N} I_t) \quad (17)$$

$$h_{(i_1 i_2 \cdots i_n) W,b}\left(X\right) = a_{i_1 i_2 \cdots i_n}^{(3)} = f\left(z_{i_1 i_2 \cdots i_n}^{(3)}\right) \quad (18)$$

Suppose we have a fixed training set $\left\{\left(X^{(1)}, Y^{(1)}\right), \ldots, \left(X^{(n)}, Y^{(n)}\right)\right\}$ of n training samples; for the entire training set, the reconstruction error function [35] of the high-order automatic coding model based on the tensor distance is defined as

$$\begin{aligned} J_{TAE}\left(\theta\right) = &\left[\frac{1}{n} \sum_{i=1}^{n} \left(\frac{1}{2} \left(h_{W,b}\left(X^{(i)}\right) - Y^{(i)}\right)^T G\left(h_{W,b}\left(X^{(i)}\right) - Y^{(i)}\right)\right)\right] + \\ &\frac{\lambda}{2} \left(\sum_{p=1}^{J_1 \times J_2 \times \cdots \times J_n} \sum_{i_1}^{I_1} \sum_{i_2}^{I_2} \cdots \sum_{i_n}^{I_n} \left(W_{p i_1 i_2 \cdots i_n}^{(1)}\right)^2 + \sum_{q=1}^{I_1 \times I_2 \times \cdots \times I_n} \sum_{j_1}^{J_1} \sum_{j_2}^{J_2} \cdots \sum_{j_n}^{J_n} \left(W_{q j_1 j_2 \cdots j_n}^{(2)}\right)^2\right) \end{aligned} \quad (19)$$

where $\theta\left(W^{(1)}, b^{(1)}; W^{(2)}, b^{(2)}\right)$ denotes the parameters of the high-order automatic coding model. The first term represents the average error, and the second term represents the L_2 normalization term for preventing overfitting.

B. High-Order Back-Propagation Algorithm

Our goal is to train model parameters $\theta\left(W^{(1)}, b^{(1)}; W^{(2)}, b^{(2)}\right)$ and seek the minimum value of reconstruction error function $J_{TAE}\left(\theta\right)$. To achieve this goal, we initialize each parameter to a small random value close to zero, and then apply a gradient descent algorithm, that is, the stochastic gradient descent method, for optimizing the following parameters:

$$W_{j i_1 i_2 \cdots i_n}^{(l)} = W_{j i_1 i_2 \cdots i_n}^{(l)} - \alpha \frac{\partial J_{TAE}\left(\theta; x, y\right)}{\partial W_{j i_1 i_2 \cdots i_n}^{(l)}} \quad (20)$$

$$b^{(l)}_{i_1 i_2 \cdots i_n} = b^{(l)}_{i_1 i_2 \cdots i_n} - \alpha \frac{\partial J_{TAE}(\theta; x, y)}{\partial b^{(l)}_{i_1 i_2 \cdots i_n}} \tag{21}$$

where $l = 1, 2$, and α is the learning rate. The key step of the BP algorithm is to calculate the above partial derivatives. We will now extend the BP algorithm to the high-dimensional space (HBP) for computing the partial derivatives. This algorithm provides an efficient way of computing these partial derivatives.

The key to HBP is to obtain the partial derivatives $\frac{\partial J_{TAE}(\theta; x, y)}{\partial W^{(l)}_{j i_1 i_2 \cdots i_n}}$ and $\frac{\partial J_{TAE}(\theta; x, y)}{\partial b^{(l)}_{i_1 i_2 \cdots i_n}}$ of the reconstruction error function $J_{TAE}(\theta)$ for the parameter $\theta\left(W^{(1)}, b^{(1)}; W^{(2)}, b^{(2)}\right)$. Here, we provide the partial derivative of the overall cost function to the parameter:

$$\frac{\partial J_{TAE}(\theta; x, y)}{\partial W^{(l)}_{j i_1 i_2 \cdots i_n}} = \left[\frac{1}{n}\sum_{i=1}^{n} \frac{\partial J_{TAE}\left(\theta; x^{(i)}, y^{(i)}\right)}{\partial W^{(l)}_{j i_1 i_2 \cdots i_n}}\right] + \lambda W^{(l)}_{j i_1 i_2 \cdots i_n} \tag{22}$$

$$\frac{\partial J_{TAE}(\theta)}{\partial b^{(l)}_{i_1 i_2 \cdots i_n}} = \frac{1}{n}\sum_{i=1}^{n} \frac{\partial J_{TAE}\left(\theta; x^{(i)}, y^{(i)}\right)}{\partial b^{(l)}_{i_1 i_2 \cdots i_n}} \tag{23}$$

Similar to the BP algorithm, HBP computes the partial derivatives through two key steps: forward-pass and BP. The forward-pass stage computes the output value of $z^{(2)}, z^{(3)}, a^{(2)}, a^{(3)}$; in the BP phase, the residual of each neuron $\eta^{(3)}_i$ is first calculated with the following details:

$$\begin{aligned}
\eta^{(3)}_i &= \frac{\partial}{\partial z^{(3)}_i} J(\theta; x, y) = \frac{\partial T_i}{\partial z^{(3)}_i} = \frac{\partial}{\partial z^{(3)}_i}\left(\frac{1}{2} g_{ii}\left(f\left(z^{(3)}_i\right) - y_i\right)\right)^2 \\
&+ \frac{1}{2}\sum_{j=1, j\neq i}^{I_1\times\cdots\times I_N} g_{ij}\left(f\left(z^{(3)}_i\right) - y_i\right)\left(f\left(z^{(3)}_j - y_j\right)\right) \\
&= g_{ii}\left(f\left(z^{(3)}_i\right) - y_i\right)\cdot f'\left(z^{(3)}_i\right) + \sum_{j=1, j\neq i}^{I_1\times I_2\times\cdots\times I_N} g_{ij}\left(f\left(z^{(3)}_j\right) - y_j\right)\cdot f'\left(z^{(3)}_i\right) \\
&= f'\left(z^{(3)}_i\right)\cdot \sum_{j=1}^{I_1\times I_2\times\cdots\times I_N} g_{ij}\left(f\left(z^{(3)}_j\right) - y_j\right) \\
&= \left(a^{(3)}_i \cdot \left(1 - a^{3}_i\right)\right)\cdot \sum_{j=1}^{I_1\times I_2\times\cdots\times I_N} g_{ij}\left(a^{(3)}_i - y_j\right)
\end{aligned} \tag{24}$$

Then, we calculate the residual $\eta^{(2)}_{j_1 j_2 \cdots j_n}$ for each neuron in the hidden layer, as follows:

$$\begin{aligned}
\eta^{(2)}_{j_1 j_2 \cdots j_n} &= \sum_{i_1=1}^{I_1}\sum_{i_2}^{I_2}\cdots\sum_{i_n}^{I_n} \frac{\partial J(\theta; x, y)}{\partial z^{(3)}_{i_1 i_2 \cdots i_n}} \cdot \frac{\partial z^{(3)}_{i_1 i_2 \cdots i_n}}{\partial z^{(2)}_{j_1 j_2 \cdots j_n}} \\
&= \left(\sum_{i_1=1}^{I_1}\sum_{i_2}^{I_2}\cdots\sum_{i_n}^{I_n} w^{(2)}_{\lambda j_1 j_2 \cdots j_n} \eta^{(3)}_{i_1 i_2 \cdots i_n}\right)\left(f'\left(z^{(2)}_{j_1 j_2 \cdots j_n}\right)\right) \\
&\left(\lambda = i_n + \sum_{j=1}^{N-1}(i_j - 1)\prod_{t=j+1}^{N} I_t\right)
\end{aligned} \tag{25}$$

Next, we calculate the residual term for each neuron of the hidden and output layers:

$$\frac{\partial z^{(3)}_{k=i_1 i_2 \cdots i_n}}{\partial w^{(2)}_{\lambda j_1 j_2 \cdots j_n}} = a^{(2)}_{\lambda j_1 j_2 \cdots j_n}\left(\lambda = i_n + \sum_{j=1}^{N-1}(i_j - 1)\prod_{t=j+1}^{N} I_t\right) \tag{26}$$

$$\frac{\partial z^{(3)}_{k=j_1 j_2 \cdots j_n}}{\partial b^{(1)}_{\nu i_1 i_2 \cdots i_n}} = a^{(1)}_{\nu i_1 i_2 \cdots i_n}\left(\nu = j_n + \sum_{i=1}^{N-1}(j_i - 1)\prod_{t=i+1}^{N} J_t\right) \tag{27}$$

Finally, according to the multiplicative chain rule, the partial derivative of the reconstruction error function to the parameter is calculated:

$$\begin{aligned}
&\frac{\partial}{\partial W^{(l)}_{\mu t_1 t_2 \cdots t_n}} J(\theta; x, y) = \frac{\partial}{\partial z^{(l+1)}_{s_1 s_2 \cdots s_n}} J(\theta; x, y)\cdot \frac{\partial z^{(l+1)}_{s_1 s_2 \cdots s_n}}{\partial W^{(l)}_{\mu t_1 t_2 \cdots t_n}} = a^{(l)}_{t_1 t_2 \cdots t_n}\eta^{(l+1)}_{s_1 s_2 \cdots s_n} \\
&\left(\mu = \left(s_n + \sum_{j=1}^{N\text{-}1}\prod_{t=i+1}^{N} S_t\right)\right)
\end{aligned} \tag{28}$$

$$\frac{\partial}{\partial b^{(l)}_{s_1 s_2 \cdots s_n}} J(W, b; x, y) = \eta^{(l+1)}_{s_1 s_2 \cdots s_n} \tag{29}$$

The receiver operating characteristic (ROC) curve is used in this paper. It can be seen more intuitively that the proposed method achieves a better recognition rate. The accuracy of the experimental results can be observed through illustrations. The superiority of the proposed algorithm compared with other algorithms in classification performance can be judged with naked eyes. The ROC curve is different from the conventional diagnostic experiment evaluation method. It can divide the experimental results into multiple sequential classifications according to actual conditions, allowing intermediate states, and the current application scope is extensive. In the ROC curve, the true positive rate is used as the ordinate and the false positive rate is used as the abscissa, providing an intuitive comparison between different tests under a common scale. The ROC curve is relatively convex and is close to the upper left corner. It shows that the better the recognition effect is, the better the comparison between different indicators; the area under the curve can also evaluate the recognition effect [36,37].

C. *Tensor Deep Learning Model*

In this study, several high-order auto-encoder models are stacked to form a TDL model. The training of the TDL model includes two steps: pre-training and fine-tuning. The pre-training process is a layer-by-layer training from bottom to top, when limiting ownership values and offsets to a certain parameter space, preventing random initialization from occurring and reducing the quality factor of each hidden layer, and using unsupervised layer-by-layer greedy pre-training [38,39]. First, the original data are inputted for training the first layer and further obtaining the first hidden layer features. Second, the first layer parameters are fixed, and then the first hidden layer is used as the input. Next, the second layer is trained, and the second hidden layer features are obtained repeatedly until all layers of the depth calculation model are trained, as shown in fig. 2. This pre-training process can be used for feature fusion of heterogeneous data.

After completing the pre-training, the supervised pre-training algorithm could be used to fine-tune the parameters of the TDL model according to the class attribute label of each data object, thereby obtaining the final parameters of the TDL model. The above training process uses random training. The random training method makes the model randomly select one sample at a time to update the model parameters. Different from the traditional batch training method, all training samples are used. The random training method speeds up the training of the model and can be used for online learning to achieve better recognition result.

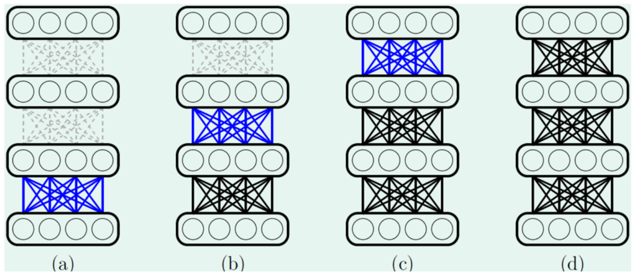

Fig.2 Layer-by-layer Greedy Pre-training

IV. Experiments

In order to estimate the validity of the proposed algorithm, two different real datasets, namely CUAVE [40] and STL-10 [41], are adopted for experiments. In the experiment, the proposed TDL model is compared with the stacked auto-encoder (SAE) model [42] and the MDL [17] based on the recognition rate. In the three models for comparison in this study, the deep learning model of our own SoftMax classifier is used, as shown in fig. 3. For fairness in comparison, each model includes two hidden layers, and the number of hidden neurons in each layer is the same. The experiment is performed on a server with 8 cores and 40 threads, Intel Xeon E5-2620 CPU, a frequency of 2.2 GHz, and a memory capacity of 64 GB. Experiments are performed on MatlabR2014 and a deep learning framework based on TensorFlow [43].

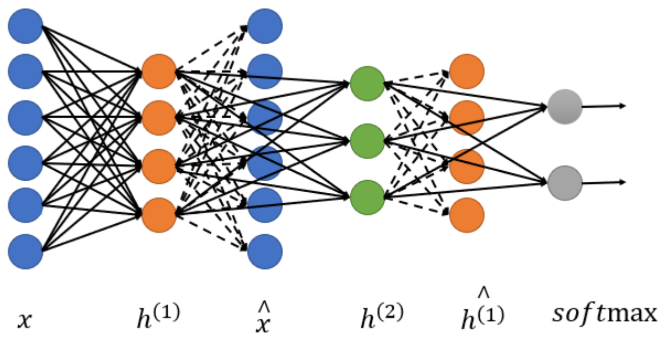

Fig.3 Feature Classification

CUAVE: The CUAVE dataset includes 36 volunteers, where each volunteer reads 10 digits from 0 to 9 five times to form a multimodal speech classification dataset. Fig. 4 shows some examples of CUAVE data. CUAVE is a multimodal dataset composed of two typical heterogeneous data, including two modes of audio and images. In the tensor-based data representation model, each instance of CUAVE is represented by a 3-order tensor as $R^{75\times50\times534}$, with 75×50 as the resolution of the image, and 534 as the audio.

STL-10: The STL-10 is a picture database that has been modified from the CIFAR10 dataset. It has formed training samples that are less marked than CIFAR10 in each category. Currently, it is widely used in deep learning, unsupervised machine learning, and other image recognition fields. It consists of 5000 training data images and 8000 test data images, which are divided into 10 categories and an additional collection. The extra set consists of 100,000 unlabeled images that are extracted from a similar but more extensive image distribution for unsupervised learning [44]. Each image can be represented by a third-order tensor $R^{96\times96\times3}$, of which 96×96 represents the resolution of the image, and 3 represents the red, green, and blue channels of the color image. Fig. 5 shows some examples of STL-10 data.

Fig.4 Sample Speakers from CUAVE Database

Fig.5 Sample from STL-10 Database

A. *Performance analysis of Tensor Factorization*

The tensor factorization is applied based on High-order singular value decomposition (HOSVD) to extract features from two-real datasets, thereby reducing the dimension of each order tensor to remove redundancy and find more meaningful latent variables. Following equation (8), the core tensor is extracted to achieve feature extraction.

A tensor toolbox [45] is used to establish our tensor factorization. In this study, our feature extraction performance is measured using the compression rate and information loss. Information loss [46, 47] is

$$IL = \frac{\left\| A - \hat{A} \right\|}{\left\| A \right\|} \tag{30}$$

The compression rate is expressed as follows:

$$CR = \frac{I_1 I_2 \cdots I_N \,\mathrm{m}}{R_1 R_2 \cdots R_N m + I_1 R_1 + I_2 R_2 + \cdots + I_N R_N} \tag{31}$$

where $I_1 I_2 \cdots I_N$ represents the original data dimension, $R_1 R_2 \cdots R_N$ represents the feature extraction dimension after dimension reduction, and m represents the entire sample size of the dataset.

For simplicity, in this study, the reduced dimension is set by setting R as the common value in our experiment. The results are presented in table 1. For the CUAVE dataset, it can be observed that the compression ratio for $40 \times 40 \times 200$ can reach 616, while most of the information from the original data are maintained (99%), which significantly reduces the memory and disk space, and is very important in the analysis of large data. For the STL-10 dataset reduced to $20 \times 20 \times 3$ in dimension, the compression ratio is as high as 2303, but the information loss is still extremely low. It can be observed that low-dimensional features of the original data can be completely obtained based on tensor factorization. In the experiments in this study, $40 \times 40 \times 200$ and $40 \times 40 \times 3$ are chosen as the low-dimensional representations of CUAVE and STL-10 datasets respectively, for the next deep learning model.

TABLE 1
PERFORMANCE OF TENSOR FACTORIZATION ON CUAVE AND STL-10

Data Set	Reduced Dimension	Compression Ratio	Information Loss
CUAVE	$20 \times 20 \times 100$	4732	4%
	$40 \times 40 \times 200$	616	1%
	$50 \times 50 \times 300$	264	0.53%
STL-10	$20 \times 20 \times 3$	2303	8.19%
	$40 \times 40 \times 3$	576	4.13%
	$50 \times 50 \times 3$	368	0.85%

B. *Performance on CUAVE*

In this experiment, first, unsupervised pre-training is performed using the Stanford multimodal dataset [15], and then, the odd-numbered data objects read by volunteers as training data are used to supervise fine-tuning of model parameters. Finally, the even data objects spoken by volunteers are adopted as test datasets to verify the correct classification rate of the model. All experiments were run five times, and the average experimental results of the five experiments are presented in table 2.

The MDL model extracts the features of audio and images separately, splices the features of the two to form a joint feature, and uses the joint features to classify the datasets. In this study, this model uses tensors to associate images with audio features at the data surface. Then, the joint features of the two are extracted, and the data are classified using their joint features. The experimental results show that the proposed algorithm is superior to the MDL model. Although the MDL model uses the joint features of two modalities for classification, it learns the characteristics of each data separately and ignores the data of underlying association. The model and SAE are also adopted to extract the audio features and image features respectively, and classify the datasets using audio features and image features respectively. The experimental results show that when using only one modality to classify the datasets, the correct rate of classification is obviously lower than the correct rate of classifying datasets with two features. In this case, it can be observed that the combination of the two features can better express each data object in the dataset. Besides, when only using image features to classify datasets, the correct rate is significantly lower than that for classifying datasets using only audio features. This shows that for the CUAVE dataset, the audio features of the dataset contribute more to the classification rate than the image features.

This study also uses the ROC curve to more intuitively analyze the experimental results. We can conclude that the algorithm has a higher accuracy. The superiority of the TDL model can be observed more clearly in fig. 6. Additional extracted heterogeneous data fusion features are shown in fig. 7.

TABLE 2
CLASSIFICATION ACCURACY ON CUAVE

Model	Classification accuracy
Stacked Auto-Encoder with visual-only (SAE)	54.3%
Stacked Auto-Encoder with audio-only (SAE)	82.1%
Tensor Deep Learning with visual-only (TDL)	73.4%
Tensor Deep Learning with audio-only (TDL)	86.7%
Tensor Deep Learning model (TDL)	96.2%
Multimodal Deep Learning (MDL)	88.5%

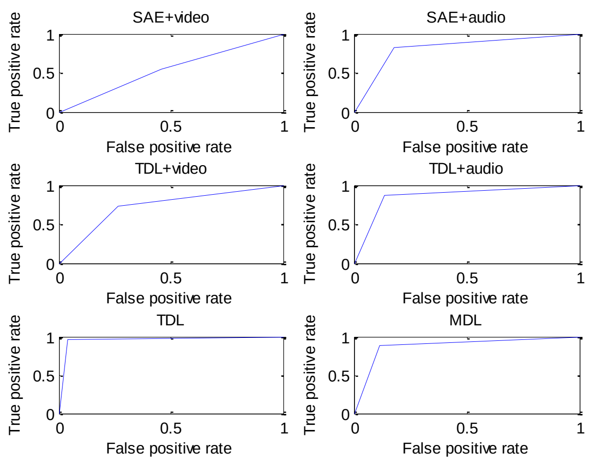

Fig.6 ROC of STL-10 Dataset

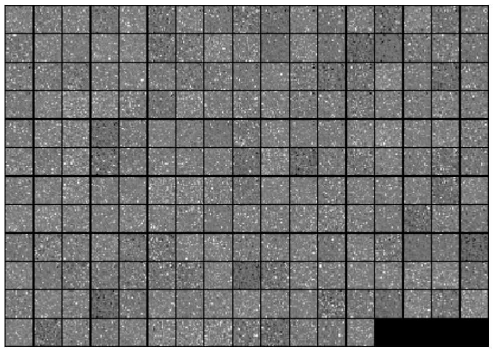

Fig.7 Fusion Features of CUAVE Dataset

C. *Performance on STL-10*

In this study, numerous datasets are also adopted for deep learning image classification to verify our proposed model. Our model can not only achieve good results in heterogeneous data fusion but it can also surpass other deep learning models in general large datasets.

The same training and testing protocols are followed to train our model. The process can be divided into three phases: (1) unsupervised training of unmarked sets, (2) supervised training of mark data from training sets, and (3) verification of the classification accuracy of models with test sets. From Section 4.1, it can be observed that the third-order tensor $R^{40\times40\times3}$ is taken as the feature of each image, whereas the SAE model needs to learn a vector with 4800 features for each hidden layer representation. All experiments were run five times, and the average experimental results of the five experiments are presented in table 3 and fig. 8. The experimental results clearly show that our model performed better than SAE, and we achieved a classification accuracy of 93.4%, which is higher than that of SAE. In this case, it is proven that the tensor space-based deep learning model is superior to the vector space-based deep learning model.

TABLE 3
CLASSIFICATION ACCURACY ON STL-10

Model	Classification accuracy
Stacked Auto-Encoder (SAE)	85.7%
Tensor Deep Learning model (TDL)	93.4%

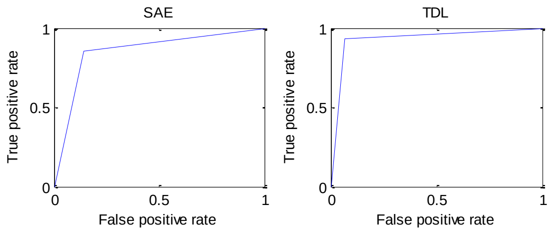

Fig.8 ROC of STL-10 Dataset

V. DISCUSSION AND CONCLUSION

Compared with a single-data source, multi-source heterogeneous data fusion significantly improves the accuracy of dataset recognition and makes the system more reliable. It can also be applied to various environments and it can ensure normal operation under severe conditions. In this study, the tensor is first used to uniformly represent multi-source heterogeneous data, which can effectively preserve the structural characteristics of the original sample and fully consider the correlation and complementarity between different modalities, so that the extracted data features can retain data. Effective information eliminates errors to ensure reliability of the system in achieving the final information optimization. Then, the tensor factorization is used to extract the core tensor to achieve data dimension reduction. The experimental results also show that the tensor factor decomposition feature can completely preserve the original information when the compression rate is particularly high, saving memory space while still obtaining most of the original data. Finally, the extracted features are adopted in the TDL model. In this study, the tensor distance instead of the Euclidean distance or cross-entropy distance is employed to reconstruct the error function and extend the conventional BP algorithm from linear space to multiple linear space. The space constitutes the HBP, which is used to fine-tune the data; thus, multi-source heterogeneous data based on feature level fusion is superior to other conventional deep learning models. Finally, the experimental results on two real datasets also show that this model can not only achieve good results on multi-source heterogeneous data, but also implement classification on large datasets quickly and achieve a relatively high recognition rate. In future research, we will determine how to eliminate the fusion of redundant data and the feature fusion of dynamic data stream.

ACKNOWLEDGMENT

This paper is supported by Natural Youth Science Foundation of China (61501326,61401310), the National Natural Science Foundation of China (61731006) and Natural Science Foundation of China (61271411). It also supported by Tianjin Research Program of Application Foundation and Advanced Technology (15JCZDJC31500), and Tianjin Science Foundation(16JCYBJC16500). This work was supported by the Tianjin Higher Education Creative Team Funds Program.

REFERENCES

[1] Chen X., Lin X., "Big Data Deep Learning: Challenges Perspectives," IEEE Access,2014. (2):514-525.
[2] D.O'Leary, "Artificial intelligence and big data," IEEE Intell. Syst., vol.28, no.2, pp.96-99, Mar.2013.
[3] *Atrey P K, Hossain M A, El Saddik A,et al , "Multimodal fusion for multimedia analysis: a survey," Multimedia systems,2010,16(6):345-379.*
[4] Jianhua Jiang, Niansong Hong, Guangyun Zhang, "A multi-source heterogeneous data fusion method and its application," Electronic Design Engineering, 2016, 24 (12):33-36.
[5] Wold S, Esbensen K, Geladi P., "Principal component analysis," Chemometrics & Intelligent Laboratory Systems, 1987, 2(1):37-52.
[6] Bai J, Wu Y, Wang G, et al, "A Novel Intrusion Detection Model Based on Multi-layer Self-Organizing Maps and Principal Component Analysis," 2006, 3973:255-260.
[7] Blei D M, Ng A Y, Jordan M I, "Latent Dirichlet Allocation," Journal of Machine Learning Research Archive, 2003, 3:993-1022.
[8] Nakamura E R, Loureiro A A F, Frery A C, "Information fusion for wireless sensor networks: Methods, models and classifications," ACM Computer SURV,2007,39 (A9):1-55.
[9] Wenchuang Li, Yongping Zhang, Yuchun Pan, "Multi-source heterogeneous data fusion model in mobile geographic information system," Journal of Computer Applications,2012,32(9):2672-2678.
[10] Tianning Yuan, "Research and implementation of multiple source heterogeneous data fusion method for EMUs," Beijing Jiaotong University,2017.
[11] Ye J, Chen K, Wu T, et al, "Heterogeneous data fusion for alzheimer's disease study," ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2008:1025-1033.
[12] Yun Ling, "Research on Heterogeneous Sensor Data Fusion Method Based on Internet of things," Computer Simulation,2011,28(11):138-140.
[13] Tiezhu Zhang, Hong Jiang, "Intelligent target recognition based on the integration of airborne radar and infrared data," Infrared and Laser Engineering,2010,39(4):756-760.
[14] Guoqing Li, Nailian Hu, Yumin Chen, "Research on multi-source heterogeneous data fusion technology in digital mining," China Mining Magazine,2011,20(4):90-93.
[15] Yongli Hu, Xinglin Pu, Yanfeng Sun, et al, "Multi-source heterogeneous sensing data fusion method and its application in target location tracking," Chinese Science,2013,43(10):1288-1306
[16] J.Ngiam, A.Khosla, M.Kim, J.Nam, H.Lee and A.Y.Ng, "Multimodal deep learning," Pro.28th Int. Conf. Mach.Learn.,2011, pp. 689-696.
[17] N. Srivastava and R.Salakhutdinov, "Multimodal learning with deep Boltzmann machines," Proc. Adv. Neural Inf. Process. Syst., 2012, pp.2222-2230.
[18] A. Cichocki, "Era of big data processing: A new approach via tensor networks and tensor decompositions," arXiv preprint arXiv: 1403-2048, 2014.
[19] D. E. Rumelhart, G. E. Hinton, and R. J. William, "Learning representations of Back-propagation errors," Nature, vol. 323, pp. 533–536, 1986.
[20] M. Vasilescu，D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," In Proceedings of the European Conference on Computer Vision (ECCV), pages 447–460, 2002.
[21] Brazell M, Li N, Navasca C, et al, "Solving multilinear systems via tensor inversion," Siam Journal on Matrix Analysis & Applications, 2013, 34(2):542-570.
[22] Zhang Q, Yang L T, Chen Z, "Deep Computation Model for Unsupervised Feature Learning on Big Data," IEEE Transactions on Services Computing, 2016, 9(1):161-171.
[23] Kolda T G, Bader B W, "Tensor Decompositions and Applications," Siam Review, 2009, 51(3):455-500.
[24] Bingyi Zhou, "Decomposition of tensor and its application in dynamic texture," Shijiazhuang: Hebei Normal University, 2010.
[25] Yan Liwei, "Research on unified expression and dimension reduction method of big data based on tensor," Huazhong University of Science and Technology, 2016.
[26] Stigant S A, "Introduction to Vector and Tensor Analysis. Introduction to vector and tensor analysis," Wiley, 1963:1119-1120.
[27] Lathauwer L D, Moor B D, Vandewalle J, "On the best rank-1 and rank-(R1, R2…, RN) approximation of higher-order tensor," Siam Journal on Matrix Analysis & Applications, 2000, 21(4):1324-1342.
[28] Vasilescu M A O, Terzopoulos D, "TensorTextures: multilinear image-based rendering," ACM SIGGRAPH. ACM, 2004:336-342.
[29] Sheehan B N, Saad Y, "Higher Order Orthogonal Iteration of Tensors (HOOI) and its Relation to PCA and GLRAM," 2007:196-199.
[30] Liu Y, Liu Y, Chan K C, "Tensor distance based multilinear locality-preserved maximum information embedding," IEEE Transactions on Neural Networks, 2010, 21(11):1848-1854.
[31] Torkhani F, Wang K, Chassery J M, "A Curvature Tensor Distance for Mesh Visual Quality Assessment," International Conference on Computer Vision and Graphics. Springer Berlin Heidelberg, 2012:253-263.
[32] Deng L, Seltzer M L, Yu D, et al, "Binary coding of speech spectrograms using a deep auto-encoder," INTERSPEECH 2010, Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September. DBLP, 2010:1692-1695.
[33] Yin X, Goudriaan J, Lantinga E A, et al, "A flexible sigmoid function of determinate growth," Annals of Botany, 2003, 91(6):753-753.
[34] Cun Y L, Boser B, Denker J S, et al, "Handwritten digit recognition with a back-propagation network," Advances in Neural Information Processing Systems, 1990, 2(2):396--404.
[35] Valle S, Weihua Li A, Qin S J," Selection of the Number of Principal Components: The Variance of the Reconstruction Error Criterion with a Comparison to Other Methods," Industrial & Engineering Chemistry Research, 1999, 38(11):653-658.
[36] Maureen O'Connell, James L.Szalma, "ROC-estimator software and roc analysis," proceedings of the HUMAN FACTORS and ERGONOMICS SOCIETY 57th ANNUAL MEETING, 2013:123-134.
[37] Wang W, Zhang M, Wang D, et al, "Kernel PCA feature extraction and the SVM classification algorithm for multiple-status, through-wall, human being detection," Eurasip Journal on Wireless Communications & Networking, 2017, 2017(1):151.
[38] Bengio Y, Lamblin P, Dan P, et al, "Greedy layer-wise training of deep networks," International Conference on Neural Information Processing Systems. MIT Press, 2006:153-160.
[39] Yu D, Seide F, Li G, "Conversational speech transcription using context-dependent deep neural networks," International Coference on International Conference on Machine Learning. Omnipress, 2012:1-2.
[40] "https://cs.stanford.edu/~acoates/stl10/"
[41] "http://people.csail.mit.edu/siracusa/avdata/"
[42] Gehring J, Miao Y, Metze F, et al, "Extracting deep bottleneck features using stacked auto-encoders," IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2013:3377-3381.
[43] "http://wiki.jikexueyuan.com/project/tensorflow-zh/"
[44] A. Coates, A. Y. Ng, and H. Lee, "An analysis of Single-layer networks in unsupervised feature learning," In Proc.Int. Conf. Artif. Intell. Statist., 2011, pp. 215–223.
[45] B. Bader and T. Kolda, "MATLAB Tensor Toolbox Version 2.2," http://csmr.ca.sandia.gov/∼tgkolda/TensorToolbox/, January 2007.

Wei Wang received his Ph.D. degree in Tianjin University, he is currently Professor in College of Electronic and Communication Engineering, Tianjin Normal University, Tianjin, China.

His main research directions involve Compressive Sensing, radar signal processing, deep learning and Multimodal data fusion and so on.

Min Zhang is currently working toward the Graduate degree in College of Electronic and Communication Engineering, Tianjin Normal University, Tianjin, China.

His research interests include deep learning, tensor space and big data.