IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 22, 2020, accepted May 29, 2020, date of publication June 2, 2020, date of current version September 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2999522

TenRR: An Approach Based on Innovative Tensor Decomposition and Optimized Ridge Regression for Judgment Prediction of Legal Cases

XIAODING GUO[1], (Member, IEEE), HONGLI ZHANG[1], LIN YE[1], SHANG LI[1], (Graduate Student Member, IEEE), AND GUANGYAO ZHANG[2]

[1]School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China
[2]Heilongjiang Branch, National Internet Emergency Response Center, Harbin 150001, China

Corresponding author: Hongli Zhang (zhanghongli@hit.edu.cn)

This work was supported by the National Key Research and Development Program of China under Grant 2018YFC0830900, Grant 2018YFC0830602, and Grant 2016QY03D0501).

ABSTRACT With the development of big data and artificial intelligence technologies, the use of computers to assist judgments in legal cases has become a popular topic. Traditional methods for judgment prediction mainly depend on feature models and classification algorithms. However, feature models require considerable expert knowledge and manual annotation work. They have strong dependence on vocabulary and grammar information in datasets, which is unconducive to improving the universality and accuracy of subsequent prediction algorithms. Meanwhile, the outputs of classification algorithms are discrete prediction results with coarse granularities. This paper proposes a new algorithm based on innovative tensor decomposition and ridge regression for judgment prediction of legal cases, namely, TenRR. TenRR is mainly divided into three steps. First, we propose a tensor representation method, namely, RTenr. RTenr expresses legal cases as three-dimensional tensors. Second, we propose an innovative tensor decomposition algorithm, namely, ITend. ITend decomposes original tensors representing legal cases into core tensors. Lastly, we propose an optimized ridge algorithm, namely, ORidge, to construct a judgment prediction model for legal cases. We further propose an optimization algorithm for ORidge to ITend; thus, core tensors obtained using ITend carry tensor elements and tensor structure information that is most beneficial to improving the accuracy of ORidge. Core tensors greatly reduce the dimension of original tensors. They eliminate the meaningless, redundant, and inaccurate information in original tensors. Experiments show that our method has higher accuracy than traditional methods for judgment prediction.

INDEX TERMS Judgment prediction, tensor decomposition, ridge regression, legal cases.

I. INTRODUCTION

With increasing maturity of big data and artificial intelligence technologies, the use of computers to assist judgments in legal cases has become a prominent research area. Judgment prediction algorithms mainly have the following two functions: (1) predict judgment results, which can provide a reference for judges, and (2) prevent the occurrence of wrongful conviction. Judgment prediction algorithms serve as a warning when the judge's judgment differs greatly from the result predicted using prediction algorithms. For example, in two burglary cases involving $30,000 each, the judgment prediction algorithm sentenced 2 years, whereas the judge sentenced 5 years.

Previous research on judgment prediction was mainly based on feature models and classification algorithms. The former is used to model legal cases. The latter predicts the scope of judgments. These methods have many shortcomings. From the perspective of feature models, (1) substantial legal expertise and manual annotation are required. These models have strong dependence on vocabulary and grammar in datasets. (2) Dimensional explosion and data sparseness are easy to occur. (3) Cases cannot be described in multiple directions. (4) Considerable inaccurate, meaningless, and

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

redundant information exists. These issues seriously affect the accuracy and stability of subsequent prediction algorithms. From the perspective of classification algorithms, (1) the granularity is coarse. Detailed prediction results cannot be provided. (2) These algorithms have a strong dependence on training data. They cannot accurately extract useful information in datasets.

This article proposes a new algorithm for judgment prediction, namely, TenRR. TenRR combines innovative tensor decomposition with optimized ridge regression. As shown in Figure 1, TenRR is mainly composed of three parts, namely, RTenr, ITend, and ORidge. First, we use RTenr to represent legal cases as three-dimensional tensors. Second, we decompose original tensors into core tensors by using ITend. Core tensors greatly reduce the dimension of original tensors. Lastly, we use core tensors to train ORidge and obtain a prediction model for legal case judgment.

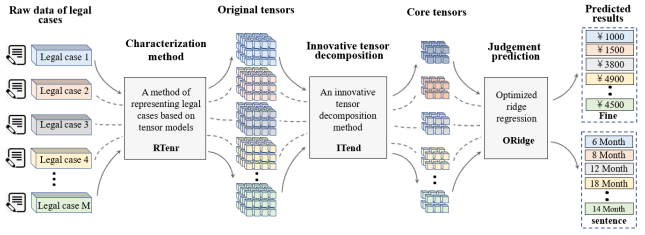


FIGURE 1. Framework of TenRR.

TenRR solves shortcomings faced by traditional judgment prediction algorithms. RTenr automatically extracts case features without the need for considerable expert knowledge and manual labeling. It characterizes cases from multiple directions. ITend greatly avoids data sparseness and dimensional explosion. It also removes substantial inaccurate, meaningless, and redundant information by using mapping matrices. ORidge is based on a regression model, which can provide fine-grained prediction results. ORidge optimizes the tensor decomposition process in ITend through mapping matrices. Therefore, obtained core tensors carry tensor elements and a tensor structure that is most beneficial to improving the accuracy of prediction methods.

The main contributions of this article are as follows:

- A method based on tensor models for representing legal cases, namely, RTenr, is proposed. RTenr represents cases as three-dimensional tensors. It automatically extracts features without a large amount of legal expertise and manual labeling. This characteristic avoids the occurrence of sparse data. RTenr has a weak dependence on lexical and grammatical information in datasets.
- An innovative tensor decomposition algorithm, namely, ITend, is introduced. ITend decomposes original tensors representing legal cases into core tensors. Core tensors greatly reduce the dimension of original tensors. ITend removes redundant, meaningless, and inaccurate information from original tensors. It improves the accuracy of subsequent prediction algorithms.
- An optimization algorithm for ITend to ORidge is recommended. ORidge uses this algorithm to guide the tensor decomposition process in ITend; hence, core tensors obtained using ITend carry tensor elements and structure information that is most conducive to improving the accuracy of ORidge.

In the remainder of this article, Section II presents research on judgment prediction of legal cases. Section III describes related calculations used in this article. Section IV details the principles of our approach. Section V provides experimental results and analysis.

II. PREVIOUS WORK

Research on judgment prediction of legal cases mainly focuses on modeling legal cases and constructing prediction methods. Previous studies mainly used feature models to describe cases and classification methods to predict judgments. Classification methods mainly include machine learning algorithms and neural networks. At present, few studies exist on judgment prediction. Considering that legal case documents are text data, this article divides previous methods into four categories on the basis of text analysis techniques, namely, (1) prediction methods based on feature models, (2) prediction methods based on matrix decomposition, (3) prediction methods based on tensor models, and (4) prediction methods based on unsupervised tensor decomposition.

Prediction methods based on feature models refer to the combination of feature models and prediction algorithms. Gruginskie [1] proposed a method based on feature models and machine learning algorithms. This method represents cases as matrices. It uses various classification algorithms, such as support vector machine and neural networks, to predict judgments. Manes and Downing [2] recommended a method based on feature models and rules. Unlike the previous method, this method uses rule reasoning to complete judgment prediction. Prediction methods based on feature models have many deficiencies, such as follows: (1) considerable expert knowledge and manual annotation are required; (2) cases cannot be described from multiple levels; and (3) data sparseness and dimensional explosion are prone to occur, which affects the accuracy and stability of subsequent prediction algorithms.

Prediction methods based on matrix decomposition refer to the combination of matrix decomposition and prediction algorithms. Jing [3] proposed a classification algorithm based on singular value decomposition (SVD). This method decomposes original matrices derived from feature models by using SVD. It uses obtained matrices to train neural networks. Similarly, Li [4] solved problems of data sparseness and dimensional explosion in feature models via matrix decomposition, which enhanced the accuracy and stability of prediction algorithms. Prediction methods based on matrix decomposition have deficiencies, including (1) the natural drawbacks of feature models and (2) poor guidance and

matrix decomposition that may lead to loss of useful information for prediction methods.

Prediction methods based on tensor models refer to the combination of tensor models and prediction algorithms. Wimalawarne [5] showed that regression and classification algorithms based on tensor models have advantages over matrix models. These methods represent cases as three-dimensional tensors and then train prediction algorithms by using such tensors. Tensor models automatically extract case elements. However, prediction methods based on tensor models have the following deficiencies: (1) dimensional explosion is easy to occur; (2) considerable redundant, useless, and meaningless information exists, which affects the accuracy of subsequent prediction algorithms.

Prediction methods based on unsupervised tensor decomposition refer to the combination of unsupervised tensor decomposition and prediction algorithms. Taguchi [6] reduced the dimension of original tensors via unsupervised tensor decomposition and used the obtained results as the input of prediction algorithms. Similarly, Zheng [7] proposed a method based on Tucker tensor decomposition. Prediction methods based on unsupervised tensor decomposition have poor guidance. They may cause loss of information useful for judgment prediction.

III. PRELIMINARIES

In this section, we provide a number of tensor-related calculation criteria used in this article, including the calculation rules for tensors and matrices or vectors. The identity matrix is represented by E.

Definition 1 (Trace Norm): For a matrix or tensor, its trace norm is the sum of elements on the main diagonal. That is, given a matrix M, $M \in \mathbb{R}^{I\times I}$, the trace norm of M is

$$Trace(M) = \sum_{i=1}^{I} M_{ii}$$

Given a tensor χ, $\chi \in \mathbb{R}^{I_1\times\cdots\times I_N}$, where $I_1 = I_2 = \cdots = I_N$, the trace norm of χ is

$$Trace(\chi) = \sum_{i=1}^{N} \chi_{i_1 i_2\cdots i_N}$$

where $i_1 = i_2 = \cdots = i_N$

Definition 2 (Frobenius Norm): For a vector, matrix or tensor, the value of the Frobenius norm is the square root of sum of squares of all elements. That is, given a tensor χ, $\chi \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, the square of the Frobenius norm of χ is

$$\|\chi\|_F^2 = \sum_{i_1}^{I_1}\sum_{i_2}^{I_2}\cdots\sum_{i_N}^{I_N} \chi_{i_1 i_2\cdots i_N}^2$$

Definition 3 (Tensor Vectorization): Given a tensor χ, $\chi \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, the tensor vectorization of χ is χ_{vec}, $\chi_{vec} \in \mathbb{R}^{I_1 I_2\cdots I_N}$ where

$$\chi_{vec\, i_1 i_2\cdots i_N} = \chi_{i_1 i_2\cdots i_N}$$

That is, the vectorization of a tensor refers to the vector formed by expanding all the elements of the tensor.

Definition 4 (n-Mode Matricization): Given a tensor χ, $\chi \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, n $\in [1, N]$, the n-mode matricization of χ is $\chi_{(n)}$, $\chi_{(n)} \in \mathbb{R}^{(I_1\cdots I_{n-1}I_{n+1}\cdots I_N)\times I_n}$, where

$$\chi_{(n) i_1 i_2\cdots i_N} = \chi_{i_1 i_2\cdots i_N}$$

That is, the n-mode matricization of a tensor is the n-dimensional expansion of the tensor.

Definition 5 (Hadamard Product): Given two matrices A and B, $A, B \in \mathbb{R}^{I\times J}$, the Hadamard product of A and B is

$$A{*}B = \begin{bmatrix} A_{11}B_{11} & A_{12}B_{12} & \cdots & A_{1J}B_{1J} \\ A_{21}B_{21} & A_{22}B_{22} & \cdots & A_{2J}B_{2J} \\ \vdots & \vdots & \ddots & \vdots \\ A_{I1}B_{I1} & A_{I2}B_{I2} & \cdots & A_{IJ}B_{IJ} \end{bmatrix}$$

That is, for two vectors, matrices or tensors with the same dimensions, the Hadamard product has the same dimension.

Definition 6 (n-Mode Product): Given a tensor χ and a matrix A, $\chi \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, $A \in \mathbb{R}^{I_n\times J_n}$, $n \in [1, N]$, the n-mode product of χ and A is $\lambda = \chi\times_n A$, $\lambda \in \mathbb{R}^{I_1\times\cdots\times I_{n-1}\times J_n\times I_{n+1}\times\cdots\times I_N}$, that

$$\lambda_{i_1\cdots i_{n-1} j_n i_{n+1}\cdots i_N} = \sum_{i=1}^{I_n} \chi_{i_1\cdots i_{n-1} i i_{n+1}\cdots i_N} A_{i j_n}$$

IV. METHODOLOGY

This article proposes a method based on innovative tensor decomposition and optimized ridge regression for judgment prediction of legal cases, namely, TenRR. As shown in Figure 2, TenRR is mainly composed of three modules. (1) RTenr. RTenr represents each legal case as a three-dimensional original tensor. (2) ITend. ITend decomposes the original tensor representing a legal case into a core tensor by using a set of mapping matrices. (3) ORidge. This article proposes an optimization method for ORidge with respect to the set of mapping matrices. ORidge controls the tensor decomposition process in ITend by optimizing mapping matrices. As a result, the obtained core tensors carry tensor elements and tensor structure information that is most conducive to improving the accuracy of TenRR.

A. RTenr

The principle of predicting judgments of legal cases is to model cases. Traditional case-modeling methods are based on feature models, which have the following disadvantages: (1) a large amount of legal expertise and manual labeling are required; (2) the explosion of dimensions and the problem of sparse data are easy to appear; and (3) feature models have a strong dependence on lexical and grammatical information in datasets. This situation greatly increases the computational complexity and volatility of subsequent prediction algorithms while reducing their accuracy and stability.

This article proposes a method based on tensor models for describing legal cases, namely, RTenr. RTenr represents legal

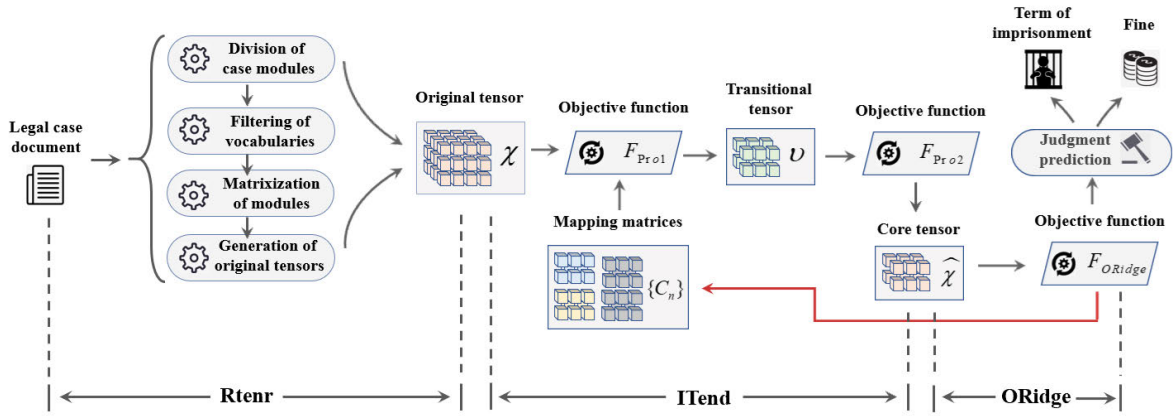


FIGURE 2. Technique route of TenRR.

cases as three-dimensional tensors. RTenr mainly includes the following steps: (1) division of case modules, (2) filtering of vocabularies in each module, (3) matrixization of modules, and (4) generation of original tensors.

Division of case modules refers to the division of legal cases into multiple modules. In accordance with previous research and expert consultation, we divide each legal case into five modules, namely, subject, object, behavior, reason, and result modules. The subject module refers to the victims in legal cases and their background information. The object module refers to the suspects in legal cases and their background information. The behavior module refers to the process of committing crimes. The reason module refers to the cause of cases and the subjective attitude of victims and suspects. The result module refers to the property loss and social effect caused by cases.

Filtering of vocabularies in each module refers to the cleaning of vocabularies in each module. All tensors representing legal cases have the same dimension; consequently, the number of vocabularies in each module is the same. Filtering of vocabularies is mainly divided into three steps.

- *Vocabulary reduction*: Dictionaries of stop words and legal terms are constructed. Meaningless and redundant vocabularies in case modules are removed. Legal-related terms are retained to avoid the loss of case elements caused by word segmentation errors.
- *Vocabulary ranking*: The frequency and TF-IDF (term frequency-inverse document frequency) value of each vocabulary in case modules are calculated. The vocabularies are sorted in accordance with the occurrence order to ensure that words that are crucial to judgment prediction of legal cases rank first.
- *Cutting or padding*: The standard length of case modules is set on the basis of the distribution of sample length in the dataset of legal cases. Modules with more vocabularies than the standard length are cut. Modules with less vocabularies than the standard length are padded.

Matrixization of modules refers to the representation of each case module as a matrix. On the basis of filtering of vocabularies, we use Google's word2vec tool and a large number of Chinese corpora to train the word vector model. Vocabularies in case modules are represented as low-dimensional dense vectors. A padded word can be represented as a zero, mean, or random vector. Generation of original tensors refers to representing legal cases as three-dimensional tensors. We merge matrices representing case modules in a three-dimensional space and then obtain an original tensor representing a legal case.

Compared with traditional feature models, RTenr has advantages. One is the automatic extraction of case features. RTenr requires no large amount of legal expert knowledge and manual annotation. It can avoid data sparsity. The other is the description of cases from various levels. It captures the potential correlation among case modules. Based on the abovementioned characteristics, RTenr greatly improves the accuracy and universality of subsequent algorithms.

B. ITend

Original tensors representing legal cases derived using RTenr cannot be directly used as inputs of subsequent judgment prediction algorithms, mainly due to the following two points. (1) When the size of legal cases is large, original tensors obtained using RTenr have high dimensions, which easily cause dimensional explosion. This phenomenon seriously increases the computational complexity of subsequent judgment prediction algorithms. (2) Original tensors obtained using RTenr carry a large amount of redundant, meaningless, and inaccurate information, which seriously affects the accuracy of subsequent judgment prediction algorithms.

This article uses the tensor decomposition strategy to solve the aforementioned problems. Tensor decomposition methods decompose original tensors into core tensors and a series of factor matrices. Core tensors represent the main tensor elements and tensor structure information of original tensors. Tensor decomposition methods have the following two advantages. (1) They greatly reduce the dimension of original tensors, thereby decreasing the computational complexity of subsequent prediction algorithms. (2) They remove the

meaningless and redundant information in original tensors and correct the inaccurate information. However, traditional tensor decomposition methods (such as CP or Tucker tensor decomposition algorithms) are unsupervised and poorly interpretable and guidable.

This article proposes an innovative tensor decomposition algorithm, namely, ITend. Unlike traditional tensor decomposition algorithms, ITend enhances interpretability by setting a set of mapping matrices. As shown in Figure 3, ITend is mainly divided into two parts. (1) Calculation of the transitional tensor. The value of the transitional tensor is calculated using original tensors and the set of mapping matrices. (2) Calculation of the core tensor. The value of the core tensor is calculated using the transitional tensor. ITend maps original tensors into a space represented by the set of mapping matrices to generate core tensors. Subsequent judgment prediction algorithms intervene in the process of tensor decomposition in ITend by optimizing the set of mapping matrices. Finally, core tensors obtained using ITend carry tensor elements and tensor structure information that is most conducive to improving the accuracy of subsequent judgment prediction algorithms.

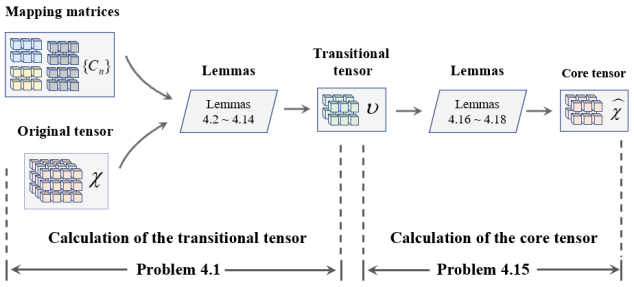


FIGURE 3. Technique route of ITend.

1) CALCULATION OF THE TRANSITIONAL TENSOR

Subsequent judgment prediction algorithms intervene in the tensor decomposition process through the set of mapping matrices in ITend. The set of mapping matrices can be interpreted as the space that is most conducive to improving the accuracy of subsequent algorithms. ITend is mainly divided into two steps: (1) calculation of the transitional tensor and (2) calculation of the core tensor. Core tensors obtained using ITend represent the tensor elements and tensor structure information that is most conducive to improving the accuracy of subsequent prediction algorithms.

In this article, the transitional tensor is calculated from the original tensor and the set of mapping matrices. The transitional tensor represents the projection of the original tensor on a space represented by the set of mapping matrices. Problem 7 provides the formal definition of the problem to be solved in this section. Under the constraints of the set of mapping matrices, the transitional tensor contains the main tensor elements and tensor structure information in the original tensor. The transitional tensor is a bridge connecting the original tensor and the set of mapping matrices. It provides a strong support for the subsequent calculation of the core tensor.

Problem 7: Given the original tensor χ derived by RTenr, χ represents a legal case, $\chi \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the set of mapping matrices $\{C_n\}$, $C_n \in \mathbb{R}^{J_n \times I_n}$, $n \in [1, N]$, calculate the value of the transitional tensor ν, $\nu \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, that ν minimize the following objective function:

$$F_{Pro1} = \left\| \chi - \nu \prod_{n=1}^{N} \times_n C_n \right\|_F^2 \tag{1}$$

where ν can be interpreted as the mapping of χ in the space represented by the set of mapping matrices $\{C_n\}$.

Lemma 8: Given a tensor χ, $\chi \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the Frobenius of χ has the following properties:

$$\|\chi\|_F^2 = \left\|\chi_{(p)}\right\|_F^2, \quad where\ p \in [1, N]$$

Lemma 9: Given a matrix A, $A \in \mathbb{R}^{I \times J}$, then the square of the Frobenius norm of A has the following properties:

$$\|A\|_F^2 = Trace(A^T A)$$

Lemma 10: Given a tensor χ and a matrix A, $\chi \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, $A \in \mathbb{R}^{I_n \times J_n}$, $n \in [1, N]$, $\lambda = \chi \times_n A$, then

$$\lambda_{(n)} = \chi_{(n)} A$$

Lemma 11: Given a matrix A, $A \in \mathbb{R}^{I \times J}$, $I \leqslant J$, there exists a B, $B \in \mathbb{R}^{J \times I}$ where B satisfies the following condition

$$AB = E$$

And the value of B can be obtained by singular value decomposition of A.

Lemma 12: Given a tensor χ and two matrices A and B, $\chi \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, $A \in \mathbb{R}^{I_n \times J_n}$, $B \in \mathbb{R}^{J_n \times K_n}$, $n \in [1, N]$, then

$$\chi \times_n A \times_n B = \chi \times_n (AB)$$

Lemma 13: Given a tensor χ and two matrices A and B, $\chi \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, $A \in \mathbb{R}^{I_n \times J_n}$, $B \in \mathbb{R}^{I_m \times J_m}$, $n, m \in [1, N]$, then

$$\chi \times_n A \times_m B = \chi \times_m B \times_n A$$

Lemma 14: Given a matrix A and a orthogonal matrix U, $A \in \mathbb{R}^{I \times J}$, $U \in \mathbb{R}^{J \times K}$, then $\|A\|_F^2 = \|AU\|_F^2 = \|UA\|_F^2$.

Lemma 15: Given a matrix A and a diagonal matrix U, $A \in \mathbb{R}^{I \times J}$, $U \in \mathbb{R}^{J \times J}$, let $\varphi(A) = \|A\|_F^2$, $\phi(A) = \|AU\|_F^2$. Then when $\phi(A)$ is minimum, $\varphi(A)$ is minimum.

Lemma 16: Given two matrices A and B, $A \in \mathbb{R}^{I \times J}$, $B \in \mathbb{R}^{J \times K}$, $B = U \sum V$, where U and V are orthogonal matrices, $\sum$ is a diagonal matrix, $U \in \mathbb{R}^{J \times J}$, $V \in \mathbb{R}^{K \times K}$, $\sum \in \mathbb{R}^{J \times K}$. Let $\varphi(A) = \|A\|_F^2$, $\phi(A) = \|AB\|_F^2$, then when $\phi(A)$ is minimum, $\varphi(A)$ is minimum.

Lemma 17: The objective function F_{Pro1} in problem 7 can be transformed into the following form:

$$F_{Pro1} = \left\| \chi \prod_{n=1}^{N} \times_n B_n - \nu \right\|_F^2 \tag{2}$$

B_n satisfies the following condition:

$$B_n = V^{(n)}\Delta^{(n)}U^{(n)T} \tag{3}$$

where $V^{(n)}$, $\Delta^{(n)}$ and $U^{(n)}$ can be obtained by performing singular value decomposition on matrix C_n. $C_n = U^{(n)}\sum^{(n)}V^{(n)T}$. $U^{(n)}$ and $V^{(n)}$ are orthogonal matrices. $\sum^{(n)}$ is a diagonal matrix. $\Delta^{(n)}$ is the inverse matrix of $\sum^{(n)}$.

According to lemma 17, we convert problem 7 into the form of equations 2 and 3. Lemmas 8, 9, 10, 11, 12, 13, 14, 15 and 16 provide support for the proof of lemma 17. Proofs 31, 32, 33, 34, 35, 36, 37, 38, 39 and 40 give the proof process of lemmas 8, 9, 10, 11, 12, 13, 14, 15, 16 and 17 respectively.

Lemma 18: Given the original tensor χ derived by RTenr, χ represents a legal case, $\chi \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, the transitional tensor ν, $\nu \in \mathbb{R}^{J_1\times J_2\times\cdots\times J_N}$, the set of matrices $\{B_n\}$, $B_n \in \mathbb{R}^{I_n\times J_n}$, $n \in [1, N]$, and the object function $F_{sub1} = Trace(v_{(p)}^T(\chi \prod_{n=1}^{N} \times_n B_n)_{(p)})$, $p \in [1, N]$, then the derivative of function F_{sub1} with respect to $\nu_{(p)}$ is

$$\frac{\partial F_{sub1}}{\partial \nu_{(p)}} = (\chi \prod_{n=1}^{N} \times_n B_n)_{(p)} \tag{4}$$

Lemma 19: Given the original tensor χ derived by RTenr, χ represents a legal case, $\chi \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, the transitional tensor ν, $\nu \in \mathbb{R}^{J_1\times J_2\times\cdots\times J_N}$, the set of matrices $\{B_n\}$, $B_n \in \mathbb{R}^{I_n\times J_n}$, $n \in [1, N]$, and the object function $F_{sub2} = Trace((\chi \prod_{n=1}^{N} \times_n B_n)_{(p)}^T \nu_{(p)})$, $p \in [1, N]$, then the derivative of function F_{sub2} with respect to $\nu_{(p)}$ is

$$\frac{\partial F_{sub2}}{\partial \nu_{(p)}} = (\chi \prod_{n=1}^{N} \times_n B_n)_{(p)} \tag{5}$$

Lemma 20: Given the transitional tensor ν, $\nu \in \mathbb{R}^{J_1\times J_2\times\cdots\times J_N}$, and the object function $F_{sub3} = Trace(v_{(p)}^T \nu_{(p)})$, $p \in [1, N]$, then the derivative of function F_{sub3} with respect to $\nu_{(p)}$ is

$$\frac{\partial F_{sub3}}{\partial \nu_{(p)}} = \xi \nu_{(p)} \tag{6}$$

where ξ is a constant, $\xi = 2$.

In this article, we use the least squares method to calculate the value of ν in equation 2. The key is to find out the partial derivative of the object function F_{Pro1} with respect to ν. Let $\varpi = \chi \prod_{n=1}^{N} \times_n B_n$, $\varpi \in \mathbb{R}^{J_1\times J_2\times\cdots\times J_N}$. Then $F_{Pro1} = \|\varpi - \nu\|_F^2$. By lemma 8, we can get that $F_{Pro1} = \left\|\varpi_{(p)} - \nu_{(p)}\right\|_F^2$, where $p \in [1, N]$. Let $C = \varpi_{(p)}$, $D = \nu_{(p)}$, $C, D \in \mathbb{R}^{(J_1J_2\cdots J_{p-1}J_{p+1}\cdots J_N)\times J_p}$. $F_{Pro1} = \|C - D\|_F^2$. According to lemma 9, $F_{Pro1} = Trace((C - D)^T(C - D))$. $F_{Pro1} = Trace(C^TC - D^TC - C^TD + D^TD)$. Since $Trace(C^TC)$ is not a function of D, $\frac{\partial Trace(C^TC)}{\partial D} = 0$. Therefore, the remaining problem is to solve the values of the following partial derivatives $\frac{\partial Trace(D^TC)}{\partial D}$, $\frac{\partial Trace(C^TD)}{\partial D}$, $\frac{\partial Trace(D^TD)}{\partial D}$.

By lemmas 18, 19 and 20, we can obtain that

$$\frac{\partial Trace(v_{(p)}^T(\chi \prod_{n=1}^{N} \times_n B_n)_{(p)})}{\partial \nu_{(p)}} = (\chi \prod_{n=1}^{N} \times_n B_n)_{(p)},$$

$$\frac{\partial Trace((\chi \prod_{n=1}^{N} \times_n B_n)_{(p)}^T \nu_{(p)})}{\partial \nu_{(p)}} = (\chi \prod_{n=1}^{N} \times_n B_n)_{(p)},$$

$$\frac{\partial Trace(v_{(p)}^T \nu_{(p)})}{\partial \nu_{(p)}} = \xi \nu_{(p)},$$

where ξ is a constant, $\xi = 2$. Proofs 41, 42 and 43 give the proof process of lemmas 18, 19 and 20, respectively. In summary, we can get that $\frac{\partial F_{Pro1}}{\partial \nu_{(p)}} = 2(\nu_{(p)} - (\chi \prod_{n=1}^{N} \times_n B_n)_{(p)})$. That is

$$\frac{\partial F_{Pro1}}{\partial \nu} = \xi(\nu - \chi \prod_{n=1}^{N} \times_n B_n) \tag{7}$$

where ξ is a constant, $\xi = 2$. According to the least squares method, we set $\frac{\partial F_{Pro1}}{\partial \nu}$ to 0. Finally, we get the calculation of ν, that is

$$\nu = \chi \prod_{n=1}^{N} \times_n B_n \tag{8}$$

2) CALCULATION OF THE CORE TENSOR

In Sub-subsection IV-B1, ITend provides the set of mapping matrices $\{C_n\}, n \in [1, N]$. $\{C_n\}$ can be interpreted as the mapping space that is most conducive to improving the accuracy of subsequent judgment prediction algorithms. We map the original tensor χ, which represents the legal case into the space represented by $\{C_n\}$, and then obtain the transitional tensor ν. ν represents the main tensor elements and tensor structure information in χ. We use the core tensor $\widehat{\chi}$ to approximate ν. Accordingly, $\widehat{\chi}$ represents the main tensor information in χ that is most conducive to improving the accuracy of subsequent judgment prediction algorithms.

Problem 21: Given the transitional tensor ν, $\nu \in \mathbb{R}^{J_1\times J_2\times\cdots\times J_N}$, calculate the value of the core tensor $\widehat{\chi}$, $\widehat{\chi} \in \mathbb{R}^{J_1\times J_2\times\cdots\times J_N}$, so that $\widehat{\chi}$ minimize the following objective function:

$$F_{Pro2} = \|\nu - \widehat{\chi}\|_F^2 \tag{9}$$

where $\widehat{\chi}$ can be interpreted as the main element information in tne original tensor χ and the main structure information in the set of mapping matrices $\{C_n\}$.

Combining equation 1 in problem 7 and equation 13 in problem 21, we can conclude that with ν as the bridge, $\widehat{\chi}$ represents both the main tensor element information in χ and the main tensor structure information in $\{C_n\}$. Therefore, $\widehat{\chi}$ is interpreted as the tensor element and tensor structure information in χ that is most conducive to improving the accuracy of the subsequent judgment prediction algorithms.

Lemma 22: Given the transitional tensor ν, $\nu \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, the core tensor $\widehat{\chi}$, $\widehat{\chi} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, and the object function $F_{sub4} = Trace(\widehat{\chi}_{(p)}^T \nu_{(p)})$, $p \in [1, N]$, then the derivative of function F_{sub4} with respect to $\widehat{\chi}_{(p)}$ is

$$\frac{\partial F_{sub4}}{\partial \widehat{\chi}_{(p)}} = \nu_{(p)} \tag{10}$$

Lemma 23: Given the transitional tensor ν, $\nu \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, the core tensor $\widehat{\chi}$, $\widehat{\chi} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, and the object function $F_{sub5} = Trace(\nu_{(p)}^T \widehat{\chi}_{(p)})$, $p \in [1, N]$, then the derivative of function F_{sub5} with respect to $\widehat{\chi}_{(p)}$ is

$$\frac{\partial F_{sub5}}{\partial \widehat{\chi}_{(p)}} = \nu_{(p)} \tag{11}$$

Lemma 24: Given the transitional tensor ν, $\nu \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, the core tensor $\widehat{\chi}$, $\widehat{\chi} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, and the object function $F_{sub6} = Trace(\widehat{\chi}_{(p)}^T \widehat{\chi}_{(p)})$, $p \in [1, N]$, then the derivative of function F_{sub6} with respect to $\widehat{\chi}_{(p)}$ is

$$\frac{\partial F_{sub6}}{\partial \widehat{\chi}_{(p)}} = \xi \widehat{\chi}_{(p)} \tag{12}$$

where ξ is a constant, $\xi = 2$.

We use the least squares method to find the value of $\widehat{\chi}$ in equation 13. According to the calculation steps of the least square method, we need to find out the partial derivative of the object function $F_{\mathrm{Pr}o2}$ with respect to $\widehat{\chi}$. By lemma 8, we can obtain that $F_{\mathrm{Pr}o2} = \left\| \nu_{(p)} - \widehat{\chi}_{(p)} \right\|_F^2$, where $p \in [1, N]$. By lemma 9, we can get that $F_{\mathrm{Pr}o2} = Trace((\nu_{(p)} - \widehat{\chi}_{(p)})^T (\nu_{(p)} - \widehat{\chi}_{(p)}))$. That is $F_{\mathrm{Pr}o2} = Trace(\nu_{(p)}^T \nu_{(p)} - \nu_{(p)}^T \widehat{\chi}_{(p)} - \widehat{\chi}_{(p)}^T \nu_{(p)} + \widehat{\chi}_{(p)}^T \widehat{\chi}_{(p)})$. Since $\nu_{(p)}$ is not a function of $\widehat{\chi}_{(p)}$, $\frac{\partial Trace(\nu_{(p)}^T \nu_{(p)})}{\partial \widehat{\chi}_{(p)}} = 0$. Based on lemmas 22, 23 and 24, we can get that $\frac{\partial Trace(\nu_{(p)}^T \widehat{\chi}_{(p)})}{\partial \widehat{\chi}_{(p)}} = \nu_{(p)}$, $\frac{\partial Trace(\widehat{\chi}_{(p)}^T \nu_{(p)})}{\partial \widehat{\chi}_{(p)}} = \nu_{(p)}$, $\frac{\partial Trace(\widehat{\chi}_{(p)}^T \widehat{\chi}_{(p)})}{\partial \widehat{\chi}_{(p)}} = \xi \widehat{\chi}_{(p)}$, where $\xi = 2$. Based on the above analysis, we can get that $\frac{\partial F_{\mathrm{Pr}o2}}{\partial \widehat{\chi}_{(p)}} = \xi(\widehat{\chi}_{(p)} - \nu_{(p)})$. That is:

$$\frac{\partial F_{\mathrm{Pr}o2}}{\partial \widehat{\chi}} = \xi(\widehat{\chi} - \nu) \tag{13}$$

where ξ is a constant, $\xi = 2$. According to the least squares method, let $\frac{\partial F_{\mathrm{Pr}o2}}{\partial \widehat{\chi}} = 0$, we can get that

$$\widehat{\chi} = \nu \tag{14}$$

Proofs 44, 45 and 46 give the proof process of lemmas 22, 23 and 24, respectively. Lemmas 22, 23 and 24 provide the theoretical support for the solution of the core tensor $\widehat{\chi}$.

C. ORidge

In terms of judgment prediction of legal cases, the dimension of the core tensor $\widehat{\chi}$ obtained using ITend is much smaller than the dimension of the original tensor χ obtained using RTenr. However, $\widehat{\chi}$ may be multicollinear. Multicollinearity has a considerable effect on judgment prediction algorithms. It can cause the following problems: (1) parameter estimates are sensitive, unstable, and distorted; (2) the variance and covariance of parameter estimates are large; and (3) the analysis function of models is reduced.

To solve the abovementioned problems, this article proposes an optimization algorithm based on ridge regression and a set of mapping matrices, namely, ORidge. Based on the traditional ridge regression model, ORidge introduces a set of mapping matrices into its loss function. This article also proposes an optimization algorithm for the loss function in ORidge versus the set of mapping matrices. ORidge controls the tensor decomposition process in ITend by optimizing the set of mapping matrices. As a result, core tensors obtained using ITend carry tensor elements and tensor structure information that is most conducive to improving the accuracy of ORidge. ORidge uses the $L2$ regular term to prevent overfitting and solve problems caused by multicollinearity. Definition 25 presents the formal definition of the loss function in ORidge.

Definition 25: Given the core tensor $\widehat{\chi}^{(m)}$, $\widehat{\chi}^{(m)} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, $m \in [1, M]$, and the set of mapping matrices $\{C_n\}$, $C_n \in \mathbb{R}^{J_n \times I_n}$, $n \in [1, N]$, $\widehat{\chi}^{(m)}$ represents a legal case. The loss function of ORidge is defined as

$$F_{ORidge} = \frac{1}{2M} \sum_{m=1}^{M} \left\| \varphi(\widehat{\chi}^{(m)}) - \phi(\widehat{\chi}^{(m)}, \mu) \right\|_F^2 + F_\psi(\alpha, \mu, \{C_n\}) \tag{15}$$

where $\varphi(\widehat{\chi}^{(m)})$ represents the judgment result of the legal case represented by $\widehat{\chi}^{(m)}$, including sentences and fines. $\phi(\widehat{\chi}^{(m)}, \mu) = \widehat{\chi}_{vec}^{(m)T} \mu$. $\mu \in \mathbb{R}^{J_1 J_2 \cdots J_N}$. $F_\psi(\alpha, \mu, \{C_n\}) = \psi(\alpha_0, \mu) + \sum_{n=1}^{N} \psi(\alpha_n, C_n)$, $\alpha \in \mathbb{R}^{N+1}$. $\psi(\lambda, A) = \lambda \|A\|_F^2$. $\|A\|_F^2$ is the $L2$ regular term.

Lemma 26: Given matrices A, B and C, $A \in \mathbb{R}^{P \times M}$, $B \in \mathbb{R}^{M \times N}$, $C \in \mathbb{R}^{N \times P}$. Then Trace(ABC) = Trace(CAB).

Lemma 27: Given a tensor $\widehat{\chi}^{(m)}$, $\widehat{\chi}^{(m)} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, $m \in [1, M]$, parameters of ORidge μ, $\mu \in \mathbb{R}^{J_1 J_2 \cdots J_N}$, $f_\varphi = Trace(\varphi(\widehat{\chi}^{(m)})^T \widehat{\chi}_{vec}^{(m)T} \mu)$. $\widehat{\chi}^{(m)}$ represents a legal case, $\varphi(\widehat{\chi}^{(m)})$ represents the judgment result of $\widehat{\chi}^{(m)}$, including sentences and fines. Then

$$\frac{\partial f_\varphi}{\partial \widehat{\chi}_{vec}^{(m)}} = \varphi(\widehat{\chi}^{(m)}) \mu$$

Lemma 28: Given a tensor $\widehat{\chi}^{(m)}$, $\widehat{\chi}^{(m)} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, $m \in [1, M]$, parameters of ORidge μ, $\mu \in \mathbb{R}^{J_1 J_2 \cdots J_N}$, $f_\chi = Trace(\mu^T \widehat{\chi}_{vec}^{(m)} \varphi(\widehat{\chi}^{(m)}))$. $\widehat{\chi}^{(m)}$ represents a legal case, $\varphi(\widehat{\chi}^{(m)})$ represents the judgment result of $\widehat{\chi}^{(m)}$, including sentences and fines. Then

$$\frac{\partial f_\chi}{\partial \widehat{\chi}_{vec}^{(m)}} = \varphi(\widehat{\chi}^{(m)}) \mu$$

Lemma 29: Given a tensor $\widehat{\chi}^{(m)}$, $\widehat{\chi}^{(m)} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_N}$, $m \in [1, M]$, parameters of ORidge μ, $\mu \in \mathbb{R}^{J_1 J_2 \cdots J_N}$, $f_T = Trace(\mu^T \widehat{\chi}_{vec}^{(m)} \widehat{\chi}_{vec}^{(m)T} \mu)$. $\widehat{\chi}^{(m)}$ represents a legal case, $\varphi(\widehat{\chi}^{(m)})$ represents the judgment result of $\widehat{\chi}^{(m)}$, including sentences

and fines. Then

$$\frac{\partial f_T}{\partial \widehat{\chi}_{vec}^{(m)}} = \xi \mu \mu^T \widehat{\chi}_{vec}^{(m)}$$

where ξ is a constant, $\xi = 2$.

Lemma 30: Given a matrix C_n, $C_n \in \mathbb{R}^{J_n \times I_n}$, $F_C = Trace(C_n^T C_n)$, then

$$\frac{\partial F_C}{\partial C_n} = \xi C_n$$

where ξ is a constant, $\xi = 2$.

We use mini-batch gradient descent (MBGD) to solve values of parameters in function 15. The key to the problem is to find out the partial derivative of function 15 with respect to C_n. Let $F_m = \left\| \varphi(\widehat{\chi}^{(m)}) - \phi(\widehat{\chi}^{(m)}, \mu) \right\|_F^2$, then $F_{ORidge} = \frac{1}{2M} \sum_{m=1}^{M} F_m + \psi(\alpha_0, \mu) + \sum_{n=1}^{N} \psi(\alpha_n, C_n)$. $\frac{\partial F_{ORidge}}{\partial C_n} = \frac{1}{2M} \sum_{m=1}^{M} \frac{\partial F_m}{\partial C_n} + \frac{\partial \psi(\alpha_0, \mu)}{\partial C_n} + \frac{\partial \sum_{n=1}^{N} \psi(\alpha_n, C_n)}{\partial C_n}$. Since $\psi(\alpha_0, \mu)$ is not a function of C_n, $\frac{\partial \psi(\alpha_0, \mu)}{\partial C_n} = 0$. According to lemma 9, $F_m = Trace(\varphi(\widehat{\chi}^{(m)})^T \varphi(\widehat{\chi}^{(m)}) - \varphi(\widehat{\chi}^{(m)})^T \widehat{\chi}_{vec}^{(m)T} \mu - \mu^T \widehat{\chi}_{vec}^{(m)} \varphi(\widehat{\chi}^{(m)}) + \mu^T \widehat{\chi}_{vec}^{(m)} \widehat{\chi}_{vec}^{(m)T} \mu)$. Since $\varphi(\widehat{\chi}^{(m)})$ is not a function of C_n, $\frac{\partial Trace(\varphi(\widehat{\chi}^{(m)})^T \varphi(\widehat{\chi}^{(m)}))}{\partial C_n} = 0$. Since $\frac{\partial F}{\partial C_n} = \frac{\partial F}{\partial \widehat{\chi}_{vec}^{(m)}} \frac{\partial \widehat{\chi}_{vec}^{(m)}}{\partial C_n}$, we find the value of $\frac{\partial F}{\partial C_n}$ by calculating $\frac{\partial F}{\partial \widehat{\chi}_{vec}^{(m)}}$. Let $f_\varphi = Trace(\varphi(\widehat{\chi}^{(m)})^T \widehat{\chi}_{vec}^{(m)T} \mu)$.

By lemma 27, we can get that $\frac{\partial f_\varphi}{\partial \widehat{\chi}_{vec}^{(m)}} = \varphi(\widehat{\chi}^{(m)})\mu$. Let $f_\chi = Trace(\mu^T \widehat{\chi}_{vec}^{(m)} \varphi(\widehat{\chi}^{(m)}))$, by lemma 28, we can get that $\frac{\partial f_\chi}{\partial \widehat{\chi}_{vec}^{(m)}} = \varphi(\widehat{\chi}^{(m)})\mu$. Let $f_T = \partial Trace(\mu^T \widehat{\chi}_{vec}^{(m)} \widehat{\chi}_{vec}^{(m)T} \mu)$, by lemma 29, we can get that $\frac{\partial f_T}{\partial \widehat{\chi}_{vec}^{(m)}} = \xi \mu \mu^T \widehat{\chi}_{vec}^{(m)}$, where ξ is a constant, $\xi = 2$. Therefore $\frac{\partial F_m}{\partial \widehat{\chi}_{vec}^{(m)}} = 2(\mu \mu^T \widehat{\chi}_{vec}^{(m)} - \varphi(\widehat{\chi}^{(m)})\mu)$. According to equations 8 and 14, we can get that $\frac{\partial \widehat{\chi}_{vec}^{(m)}}{\partial C_n} = \frac{\partial \widehat{\chi}_{vec}^{(m)}}{\partial B_n} \frac{\partial B_n}{\partial C_n} = (\chi \prod_{k=1, k \neq n}^{N} \times_k B_k) \frac{\partial B_n}{\partial C_n}$. $\frac{\partial B_n}{\partial C_n}$ can be obtained by the rule of inverse matrix derivation. Thus

$$\frac{\partial F_m}{\partial C_n} = \xi F_\mu F_N F_B \tag{16}$$

where $F_\mu = \mu \mu^T \widehat{\chi}_{vec}^{(m)} - \varphi(\widehat{\chi}^{(m)})\mu$, $F_N = \chi \prod_{k=1, k \neq n}^{N} \times_k B_k$, $F_B = \frac{\partial B_n}{\partial C_n}$, $\xi = 2$. By lemma 30, we can get that $\frac{\partial \psi(\alpha_n, C_n)}{\partial C_n} = 2\alpha_n C_n$. Thus

$$\frac{\partial F_{ORidge}}{\partial C_n} = \frac{1}{M} \sum_{m=1}^{M} F_\mu F_N F_B + 2\alpha_n C_n \tag{17}$$

Proofs 47, 48, 49, 50 and 51 give the proof process of lemmas 26, 27, 28, 29 and 30, respectively.

V. RESULTS AND ANALYSIS

A. DATA DESCRIPTION

The dataset used in this article consists of real legal cases obtained from the Chinese Referee Document Network. The dataset contains nearly 3 million legal cases involving 203 crimes. We divide each judgment of legal cases into two parts, namely, sentences and fines. On the basis of a large amount of legal expertise, in the early work of this article, we extracted features from the original dataset and then obtained the number of sentence and fine of each legal case. Such work provided effective data support for the training of judgment prediction models.

The legal cases in the dataset involve sentences ranging from 0 to 300 (in units of months) and fines ranging from 0 to 100,000 (in units of yuan). Figure 4 shows the number of sentences and fines in some legal cases of fixed-term imprisonment. We analyze legal case data and find interesting phenomena. (1) Sentences in more than 80% of legal cases are concentrated within 3 years. (2) Fines in more than 80% of legal cases are concentrated within 6000 yuan. (3) The number of legal cases sentenced to death or life imprisonment is small, 0.159% and 0.328%, respectively. (4) In nearly 50% of legal cases, the number of fines is 0 yuan.

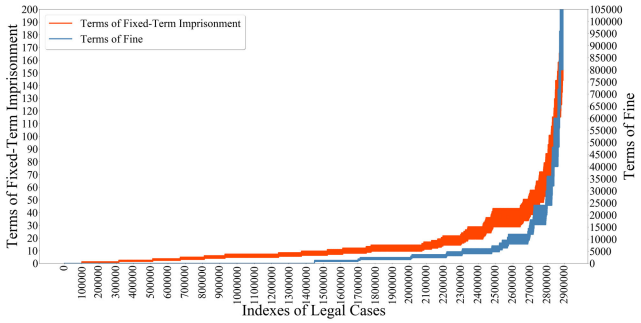


FIGURE 4. Sentences and fines of legal cases.

B. BASELINE APPROACHES

This article proposes a new method for the judgment of legal cases, namely, TenRR. TenRR mainly consists of three modules. (1) RTenr. It represents legal cases as three-dimensional original tensors. (2) ITend. It decomposes original tensors into core tensors. (3) ORidge. It is trained using the obtained core tensors, and a judgment prediction model for legal cases is obtained. The set of mapping matrices $\{C_n\}$ in ITend is crucial. $\{C_n\}$ is a bridge connecting ITend and ORidge. It enables ORidge to control the tensor decomposition process in ITend. Consequently, the obtained core tensor $\widehat{\chi}$ contains the tensor elements and tensor structure information that is most conducive to improving the accuracy of TenRR.

Studies on judgment prediction are currently limited. Proposed methods are mainly based on feature models and machine learning algorithms. Considering that the case documents are text data, we set the following four

baselines on the basis of latest research in the field of text analysis.

- *Prediction methods based on feature models:* These methods use feature models to represent legal cases. The obtained matrices are input into prediction algorithms.
- *Prediction methods based on matrix decomposition:* These methods use feature models to represent legal cases decompose original matrices via matrix decomposition, and train prediction algorithms through obtained matrices.
- *Prediction methods based on tensor models:* These methods represent legal cases as three-dimensional tensors and then input them into prediction algorithms.
- *Prediction methods based on unsupervised tensor decomposition:* These methods use tensor models to represent legal cases. They decompose original tensors into core tensors by using unsupervised tensor decomposition. Core tensors are used to train prediction algorithms.

The highlight of the method proposed in this paper is the setting of mapping matrices. Mapping matrices closely link the case-modeling process with subsequent prediction algorithms. The abovementioned four baselines can reflect the advantages of mapping matrices in TenRR. Prediction algorithms used in this article include commonly used neural networks and regression algorithms. Neural networks include TextCNN [8], TextRNN [9], TextCNN attention [10], TextRNN attention [11], LSTM [12], Bi-LSTM [13], GRU [14] and Bi-GRU [15]. Regression algorithms include linear regression [16], polynomial regression [17], ridge regression [18], Lasso regression [19], and ElasticNet regression [20].

C. PARAMETER ADJUSTMENT AND EXPERIMENTAL SETTINGS

This article proposes a new method based on the innovative tensor decomposition and optimized ridge regression, namely, TenRR. TenRR consists of three parts: RTenr, ITend, and ORidge. Unlike traditional ridge regression algorithms, in addition to the regression coefficient, the parameters involved in TenRR include a set of mapping matrices $\{C_n\}$. In TenRR, ORidge controls the tensor decomposition process in ITend by optimizing the values of $\{C_n\}$. Therefore, the selection of initial values of the set of mapping matrices $\{C_n\}$ is important. It affects the convergence speed and accuracy of judgment prediction algorithms.

Before conducting formal experiments, we establish numerous preliminary experiments on small batch datasets. We set different initial values of mapping matrices in accordance with different steps on each small batch dataset. Then, we monitor the influence of the initial values of mapping matrices on the convergence speed of prediction algorithms. From previous experiments, we can conclude that when each mapping matrix has a simple linear relationship and the absolute values of its elements are small, the loss function can reach the optimal value rapidly. That is, the prediction algorithm has a fast convergence speed.

TenRR is a regression algorithm, and each obtained sentence or fine is an accurate value. However, in practice, the judge would prefer to see the range of sentences or fines. Therefore, we set a fault tolerance window. When the difference between the predicted and real values is within the window, we consider the predicted value to be correct. In this paper, the fault tolerance window adopts a dynamic floating mechanism. For sentences, the fault tolerance window is 3 months. When the prison term is more than 10 years, the fault tolerance window can be extended up to 6 months. For fines, the fault tolerance window is 300 yuan. When the fine involved is more than 10,000 yuan, the fault tolerance window can be extended to 500 yuan. When the fine involved is more than 50,000 yuan, the fault tolerance window can be extended to 1,000 yuan.

For traditional neural networks, such as TextCNN, TextRNN, TextCNN attention, TextRNN attention, LSTM, Bi-LSTM, GRU, and Bi-GRU, this article performs 10 iterations with a batch size of 128, a hidden layer size of 512, a hidden layer number of 3, and a learning rate of 0.001. We use TensorFlow as the program development tool and a graphics processing unit to increase the calculation speed. For traditional regression algorithms, such as linear, polynomial, Lasso, and ElasticNet, this article uses the optimal order and regression coefficients in each specified interval.

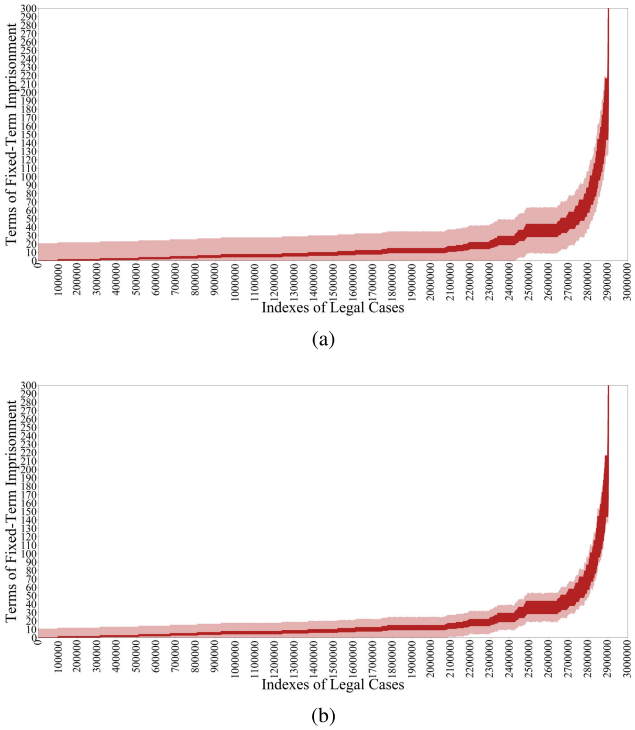
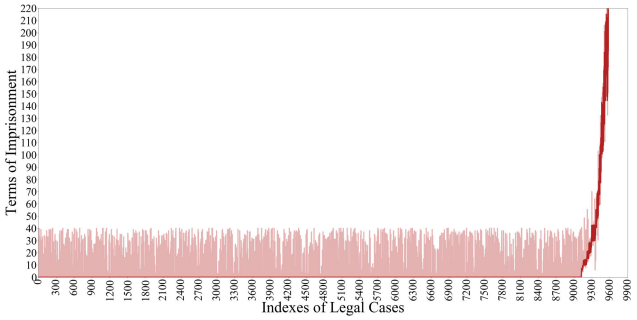


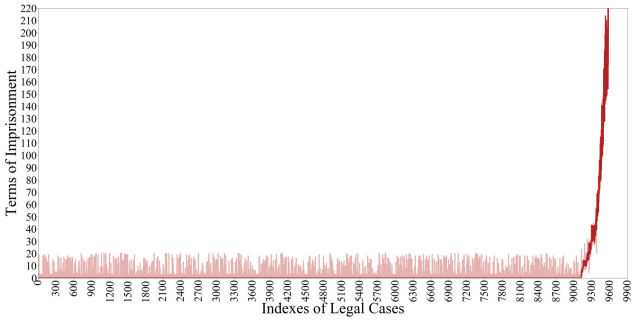
FIGURE 5. Predicted and actual values of sentences in legal cases of fixed-term imprisonment.

D. EXPERIMENTAL RESULTS AND ANALYSIS

This subsection provides the experimental results and analysis in this article. Figures 5, 6 and 7 show the predicted

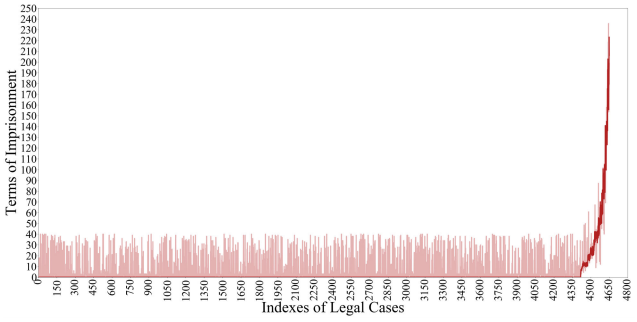


(a)

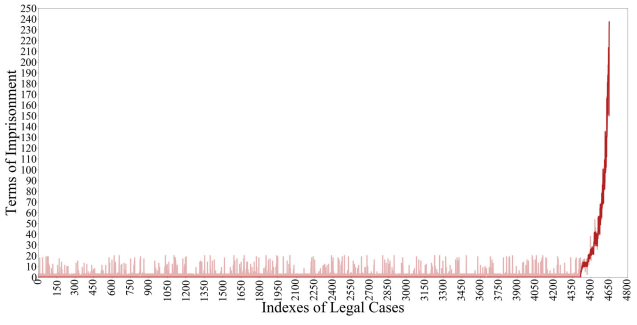


(b)

FIGURE 6. Predicted and actual values of sentences in legal cases of life imprisonment.



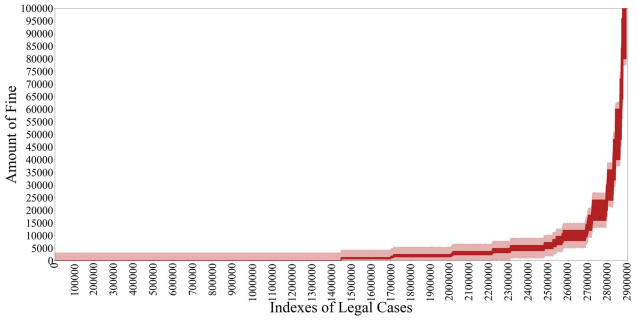
(a)



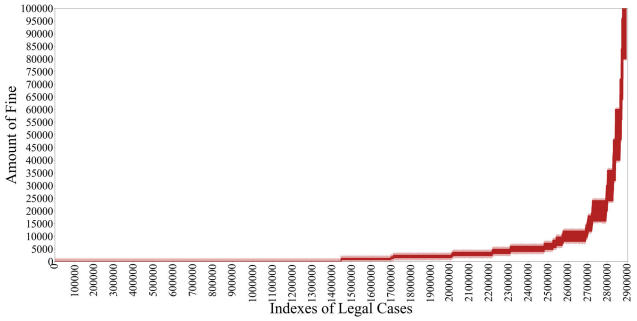
(b)

FIGURE 7. Predicted and actual values of sentences in legal cases of death penalty.

and actual values of sentences in legal cases of fixed-term imprisonment, life imprisonment, and death penalty, respectively. In cases of death penalty, sentences mean suspended sentences. In cases of life imprisonment, sentences



(a)



(b)

FIGURE 8. Predicted and actual values of fines in legal cases.

mean commutation. Figure 8 shows the predicted and actual values of the fines in legal cases. Subfigure (a) depicts the experimental results of the judgment prediction method based on tensor models, which is an algorithm combining RTenr and traditional Ridge regressions. Subfigure (b) demonstrates the experimental results of TenRR. Each figure has two curves, of which the dark curve represents the distribution of actual sentences or fines, and the light curve represents the distribution of predicted sentences or fines for corresponding legal cases.

From Figures 5, 6, 7 and 8, compared with the prediction results of the traditional ridge regression algorithms, those of TenRR on sentences and fines are more accurate. The main reason is the use of ITend and ORidge in TenRR. ITend decomposes the three-dimensional original tensor χ obtained using RTenr into a core tensor $\widehat{\chi}$. χ represents a legal case. First, ITend establishes a set of mapping matrices $\{C_n\}$. It maps χ into a feature space represented by $\{C_n\}$ and then obtains the core tensor $\widehat{\chi}$. $\widehat{\chi}$ greatly reduces the dimensions of x while removing redundant, meaningless, and inaccurate information from it. Second, ORidge intervenes in the tensor decomposition process in ITend by optimizing the value of $\{C_n\}$; therefore, the obtained core tensor $\widehat{\chi}$ carries tensor elements and tensor structure information that is most beneficial to improving the prediction accuracy of TenRR. $\{C_n\}$ is the bridge connecting ITend and ORidge. In terms of sentence prediction, the number of legal case of fixed-term imprisonment is larger than those of life imprisonment and death penalty, and, hence, the prediction accuracy rate of fixed-term imprisonment is higher.

TABLE 1. Prediction methods based on feature models.

(a) Judgment prediction based on neural networks

	Text CNN	TextCNN attention	Text RNN	TextRNN attention	LSTM	Bi-LSTM	GRU	Bi-GRU
Fixed	0.7459	0.7964	0.7731	0.8091	0.8127	0.8283	0.8137	0.8221
Life	0.7391	0.7746	0.7576	0.7875	0.8036	0.8168	0.7944	0.8025
Death	0.7348	0.7581	0.7695	0.7705	0.7928	0.8067	0.8078	0.8153
Fine	0.7209	0.7682	0.7746	0.7951	0.8074	0.8240	0.8092	0.8148

(b) Judgment prediction based on regression algorithms

	Linear regression	Polynomial regression	Ridge regression	Lasso regression	ElasticNet regression
Fixed	0.6721	0.7129	0.7430	0.7586	0.7732
Life	0.6674	0.7017	0.7245	0.7429	0.7595
Death	0.6723	0.6991	0.7239	0.7366	0.7668
Fine	0.6836	0.7294	0.7429	0.7543	0.7792

TABLE 2. Prediction methods based on matrix decomposition.

(a) Judgment prediction based on neural networks

	Text CNN	TextCNN attention	Text RNN	TextRNN attention	LSTM	Bi-LSTM	GRU	Bi-GRU
Fixed	0.7570	0.8048	0.7903	0.8185	0.8206	0.8326	0.8259	0.8298
Life	0.7541	0.7808	0.7654	0.8039	0.8109	0.8265	0.8039	0.8196
Death	0.7447	0.7604	0.7761	0.7932	0.8143	0.8278	0.8148	0.8272
Fine	0.7303	0.7742	0.7852	0.7923	0.8094	0.8165	0.8093	0.8343

(b) Judgment prediction based on regression algorithms

	Linear regression	Polynomial regression	Ridge regression	Lasso regression	ElasticNet regression
Fixed	0.6887	0.7384	0.7591	0.7614	0.7946
Life	0.6743	0.7149	0.7465	0.7678	0.7782
Death	0.6919	0.7113	0.7361	0.7550	0.7872
Fine	0.6925	0.7327	0.7530	0.7602	0.7876

TABLE 3. Prediction methods based on tensor models.

(a) Judgment prediction based on neural networks

	Text CNN	TextCNN attention	Text RNN	TextRNN attention	LSTM	Bi-LSTM	GRU	Bi-GRU
Fixed	0.7853	0.8272	0.8196	0.8325	0.8461	0.8589	0.8437	0.8609
Life	0.7704	0.8169	0.8023	0.8174	0.8335	0.8495	0.8302	0.8473
Death	0.7683	0.8092	0.7951	0.8002	0.8198	0.8346	0.8237	0.8354
Fine	0.7842	0.8214	0.8205	0.8371	0.8316	0.8533	0.8408	0.8592

(b) Judgment prediction based on regression algorithms

	Linear regression	Polynomial regression	Ridge regression	Lasso regression	ElasticNet regression	TenRR
Fixed	0.7268	0.7724	0.8087	0.8175	0.8443	**0.9371**
Life	0.7135	0.7618	0.7976	0.8029	0.8326	**0.9162**
Death	0.7123	0.7591	0.7998	0.7996	0.8259	**0.9260**
Fine	0.7256	0.7742	0.7963	0.8187	0.8424	**0.9356**

Tables 1 to 4 present the judgment prediction methods based on feature models, matrix decomposition, tensor models, and unsupervised tensor decomposition, respectively. For convenience, the prediction accuracy of TenRR is also shown in Table 3.

Subtables (a) in Tables 1 to 4 demonstrate the accuracy of judgment prediction algorithms based on neural networks. In this article, every 3 months is classified as one category. Every 100 yuan is also classified as one category. The prediction accuracy of RNNs is higher than that of CNNs.

TABLE 4. Prediction methods based on unsupervised tensor decomposition.

(a) Judgment prediction based on neural networks

	Text CNN	TextCNN attention	Text RNN	TextRNN attention	LSTM	Bi-LSTM	GRU	Bi-GRU
Fixed	0.8009	0.8381	0.8263	0.8483	0.8540	0.8606	0.8531	0.8744
Life	0.7924	0.8295	0.8206	0.8364	0.8527	0.8544	0.8439	0.8663
Death	0.7790	0.8230	0.8175	0.8304	0.8225	0.8415	0.8325	0.8517
Fine	0.8082	0.8367	0.8297	0.8413	0.8476	0.8611	0.8518	0.8609

(b) Judgment prediction based on regression algorithms

	Linear regression	Polynomial regression	Ridge regression	Lasso regression	ElasticNet regression
Fixed	0.7467	0.7814	0.8293	0.8311	0.8576
Life	0.7239	0.7850	0.8134	0.8255	0.8424
Death	0.7326	0.7743	0.8005	0.8231	0.8596
Fine	0.7319	0.7992	0.8171	0.8393	0.8498

RNNs can capture the contextual information among vocabularies. By contrast, in CNNs, the convolution kernel focuses on capturing spatial correlation information among vocabularies. Therefore, RNNs are better at processing time series data than CNNs. Nevertheless, LSTMs have higher prediction accuracy than RNNs. The main reason is that handling the long-distance dependency is difficult for RNNs. LSTMs, on the contrary, solve the problem by setting long-term state and gates. Compared with unidirectional LSTM, Bi-LSTM fully considers the context information of vocabularies and, thus, has higher prediction accuracy. GRUs and LSTMs have comparable accuracy, but GRU has faster convergence speed. GRUs pass outputs directly to the next neuron, and no output gate is needed. They also have fewer parameters than LSTMs.

Subtables (b) in Tables 1 to 4 present the accuracy of different regression algorithms. Polynomial regression has higher prediction accuracy than linear regression because polynomial regression can handle nonlinear relationships in datasets. Compared with linear and polynomial regression, ridge regression, Lasso regression, ElasticNet, and TenRR have higher prediction accuracy. Linear and polynomial regression have difficulty dealing with multicollinearity. Ridge regression, Lasso regression, ElasticNet, and TenRR solve the problem through L1 or L2 regular term. Compared with ridge regression, Lasso regression has higher accuracy. L1 normal form has the function of feature selection, which can reduce the influence of unnecessary features on the accuracy of prediction models.

Tables 1 and 2 indicate that prediction methods based on matrix decomposition models are more accurate than those based on feature models. Matrix decomposition methods (such as SVD) greatly reduce the dimensions of original matrices. They remove redundant information and predict missing data. These factors improve the accuracy of subsequent prediction algorithms. Similarly, as shown in Tables 3 and 4, the accuracy of prediction algorithms based on tensor decomposition models is higher than that of prediction algorithms based on tensor models.

Matrix or tensor decomposition algorithms can reduce the dimension and sparseness of original data. They help improve the accuracy and stability of subsequent prediction algorithms. Comparison of Tables 1 to 4 shows that prediction methods based on tensors perform better than those based on matrices. Tensor models can describe legal cases from multiple directions. Sparseness and accuracy of data in tensor models are better than those of data in matrix models. Tensor models also provide considerable data support for the training of subsequent prediction algorithms.

Tables 1 to 4 imply that compared with a series of neural networks and regression algorithms, TenRR has higher accuracy in judgment prediction of legal cases. The main reason is the use of mapping matrices. ITend sets mapping matrices and decomposes original tensors representing legal cases into core tensors under the guidance of mapping matrices. Core tensors greatly reduce the dimensions of original tensors while removing inaccurate, redundant, and meaningless information from them. ORidge intervenes in the tensor decomposition process in ITend by optimizing the value of mapping matrices. As a result, core tensors obtained using ITend represent the tensor elements and tensor structure information that is most conducive to improving the accuracy of TenRR.

VI. CONCLUSIONS

This article proposes a new method for judgment prediction of legal cases, namely, TenRR. TenRR is based on innovative tensor decomposition and optimized ridge regression. TenRR is mainly divided into three steps. (1) A method based on tensor models for representing legal cases, namely, RTenr. We use RTenr to represent legal cases as three-dimensional original tensors. (2) An innovative tensor decomposition method, namely, ITend. ITend decomposes original tensors into core tensors. (3) An optimized ridge regression algorithm, namely, ORidge. We train ORidge using obtained core tensors. Finally, a judgment prediction model for legal cases is obtained.

Compared with judgment prediction models based on feature models and classification algorithms, TenRR has the following advantages. (1) RTenr does not require considerable expert knowledge and manual labeling, and it can fully describe legal cases. (2) ITend establishes a set of mapping matrices $\{C_n\}$. It maps original tensors into a feature space represented by $\{C_n\}$ and then obtains core tensors. Core tensors greatly reduce the dimensions of original tensors while removing redundant, meaningless, and inaccurate information from them. ITend avoids dimensional explosion and sparse data. (3) ORidge interferes with the tensor decomposition process in ITend by optimizing the value of mapping matrices $\{C_n\}$. Therefore, core tensors derived using ITend carry the tensor elements and tensor structure information that is most conducive to improving the accuracy of TenRR. The aforementioned advantages greatly improve the accuracy of TenRR. This article further proposes an optimization algorithm for ORidge with respect to the set of mapping matrices $\{C_n\}$. First, we calculate the partial derivative of the loss function with respect to mapping matrices $\frac{\partial F_{ORidge}}{\partial C_n}$. Then, we complete the iteration of $\{C_n\}$ by using MBGD.

FIRST APPENDIX

Proofs 31, 32, 33, 34, 35, 36, 37, 38, 39 and 40 give the proof process of lemmas 8, 9, 10, 11, 12, 13, 14, 15, 16 and 17 respectively.

Proof 31: From the definition of tensor matrixization 4, we know that the elements in $\chi_{(p)}$ have not changed compared to χ, so the sum of the squares of their elements is equal, that is, $\left\|\chi_{(p)}\right\|_F^2 = \|\chi\|_F^2$.

Proof 32: By definition 2, $\|A\|_F^2 = \sum_{i=1}^{I}\sum_{j=1}^{J} A_{ij}^2$. Let $B = A^TA$, $B \in \mathbb{R}^{J\times J}$. According to definition 1, $Trace(B) = \sum_{j=1}^{J} B_{jj}$. From the rule of matrix multiplication, $B_{jj} = \sum_{i=1}^{I} A_{ji}^T A_{ij}$. That is $B_{jj} = \sum_{i=1}^{I} A_{ij}^2$. Therefore, $Trace(B) = \sum_{j=1}^{J} B_{jj} = \sum_{j=1}^{J}\sum_{i=1}^{I} A_{ij}^2$. $\|A\|_F^2 = Trace(B) = Trace(A^TA)$.

Proof 33: By definition 4, $\chi_{(n)} \in \mathbb{R}^{(I_1\cdots I_{n-1}I_{n+1}\cdots I_N)\times I_n}$. Let $B = \chi_{(n)}A$, $B \in \mathbb{R}^{(I_1\cdots I_{n-1}I_{n+1}\cdots I_N)\times J_n}$. $B_{(i_1i_2\cdots i_{n-1}i_{n+1}\cdots i_N)j_n} = \sum_{k=1}^{I_n} \chi_{(n)(i_1i_2\cdots i_{n-1}i_{n+1}\cdots i_N)k} A_{kj_n}$. According to the definition of 4, $\chi_{(n)(i_1i_2\cdots i_{n-1}i_{n+1}\cdots i_N)k} = \chi_{i_1i_2\cdots i_{n-1}ki_{n+1}\cdots i_N}$. Therefore, $B_{(i_1i_2\cdots i_{n-1}i_{n+1}\cdots i_N)j_n} = \sum_{k=1}^{I_n} \chi_{i_1i_2\cdots i_{n-1}ki_{n+1}\cdots i_N} A_{kj_n}$. Let $\gamma = \chi\times_n A$, from definition 6, $\gamma \in \mathbb{R}^{I_1\times\cdots\times I_{n-1}\times J_n\times I_{n+1}\times\cdots\times I_N}$. $\gamma_{i_1i_2\cdots i_{n-1}j_ni_{n+1}\cdots i_N} = \sum_{k=1}^{I_n} \chi_{i_1i_2\cdots i_{n-1}ki_{n+1}\cdots i_N} A_{kj_n}$. by definition 4, $\gamma_{(n)} \in \mathbb{R}^{(I_1\cdots I_{n-1}I_{n+1}\cdots I_N)\times J_n}$, $\gamma_{(n)(i_1i_2\cdots i_{n-1}i_{n+1}\cdots i_N)j_n} = \gamma_{i_1i_2\cdots i_{n-1}j_ni_{n+1}\cdots i_N}$. Then $\gamma_{(n)(i_1i_2\cdots i_{n-1}i_{n+1}\cdots i_N)j_n} = B_{(i_1i_2\cdots i_{n-1}i_{n+1}\cdots i_N)j_n}$. $B = \gamma_{(n)} = \chi_{(n)}A$.

Proof 34: By performing singular value decomposition on A, we can get that $A = U\sum V^T$. $U \in \mathbb{R}^{I\times I}$, $v \in \mathbb{R}^{J\times J}$, $\sum \in \mathbb{R}^{I\times J}$. U and V are orthogonal matrices. $\sum$ is the diagonal matrix, and the elements on the diagonal are not all 0, Since $I \leqslant J$, $\sum$, there exists a matrix Δ, which satisfies $\Sigma\Delta = E$. From the properties of orthogonal matrices, we can get that $UU^T = U^TU = E$, $VV^T = V^TV = E$. Let $C = V\Delta U^T$, then $AC = U\sum V^TV\Delta U^T = E$. Therefore, $B = V\Delta U^T$.

Proof 35: Let $\gamma = \chi\times_n A\times_n B$, $\tau = \chi\times_n A$. Then $\gamma = \tau\times_n B$, $\tau \in \mathbb{R}^{I_1\times\cdots\times I_{n-1}\times J_n\times I_{n+1}\times\cdots\times I_N}$, $\gamma \in \mathbb{R}^{I_1\times\cdots\times I_{n-1}\times K_n\times I_{n+1}\times\cdots\times I_N}$. From the definition of 6, $\gamma_{i_1i_2\cdots i_{n-1}k_ni_{n+1}\cdots i_N} = \sum_{j=1}^{J_n} \tau_{i_1i_2\cdots i_{n-1}ji_{n+1}\cdots i_N} B_{jk_n}$. $\tau_{i_1i_2\cdots i_{n-1}ji_{n+1}\cdots i_N} = \sum_{i=1}^{I_n} \chi_{i_1i_2\cdots i_{n-1}ii_{n+1}\cdots i_N} A_{ij}$. Then $\gamma_{i_1i_2\cdots i_{n-1}k_ni_{n+1}\cdots i_N} = \sum_{j=1}^{J_n}\sum_{i=1}^{I_n} \chi_{i_1i_2\cdots i_{n-1}ii_{n+1}\cdots i_N} A_{ij} B_{jk_n}$.

Let $C = AB$, $\nu = \chi\times_n C$, then $C \in \mathbb{R}^{I_n\times K_n}$. $\nu \in \mathbb{R}^{I_1\times\cdots\times I_{n-1}\times K_n\times I_{n+1}\times\cdots\times I_N}$. From the definition of 6,

$v_{i_1 i_2 \cdots i_{n-1} k_n i_{n+1} \cdots i_N} = \sum_{i=1}^{I_n} \chi_{i_1 i_2 \cdots i_{n-1} i i_{n+1} \cdots i_N} C_{i k_n}$. $C_{i k_n} = \sum_{j=1}^{J_n} A_{ij} B_{j k_n}$. Then $v_{i_1 i_2 \cdots i_{n-1} k_n i_{n+1} \cdots i_N} = \sum_{i=1}^{I_n} \sum_{j=1}^{J_n} \chi_{i_1 i_2 \cdots i_{n-1} i i_{n+1} \cdots i_N} A_{ij} B_{j k_n}$. Therefore, $\gamma = v$. That is $\chi \times_n A \times_n B = \chi \times_n (AB)$.

Proof 36: Assume that the value of m is less than n, let $\gamma = \chi \times_n A \times_m B$, $\tau = \chi \times_n A$, then $\gamma = \tau \times_m B$. $\gamma \in \mathbb{R}^{I_1 \times \cdots \times I_{m-1} \times J_m \times I_{m+1} \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N}$, $\tau \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N}$. By the definition of tensor product 6, we can obtain that

$$\gamma_{i_1 i_2 \cdots i_{m-1} j_m i_{m+1} \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{j=1}^{I_m} \tau_{i_1 i_2 \cdots i_{m-1} j i_{m+1} \cdots i_{n-1} j_n i_{n+1} \cdots i_N} B_{j j_m}.$$

Since

$$\tau_{i_1 i_2 \cdots i_{m-1} j i_{m+1} \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i=1}^{I_n} \chi_{i_1 i_2 \cdots i_{m-1} j i_{m+1} \cdots i_{n-1} i i_{n+1} \cdots i_N} A_{i j_n},$$

then

$$\gamma_{i_1 i_2 \cdots i_{m-1} j_m i_{m+1} \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{j=1}^{I_m} \sum_{i=1}^{I_n} \chi_{i_1 i_2 \cdots i_{m-1} j i_{m+1} \cdots i_{n-1} i i_{n+1} \cdots i_N} A_{i j_n} B_{j j_m}.$$

Let $\omega = \chi \times_m B \times_n A$, $\kappa = \chi \times_m B$, then $\omega = \kappa \times_n A$. $\omega \in \mathbb{R}^{I_1 \times \cdots \times I_{m-1} \times J_m \times I_{m+1} \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N}$, $\kappa \in \mathbb{R}^{I_1 \times \cdots \times I_{m-1} \times J_m \times I_{m+1} \times \cdots \times I_N}$. By the definition of tensor product 6, we can get that

$$\omega_{i_1 i_2 \cdots i_{m-1} j_m i_{m+1} \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i=1}^{I_n} \kappa_{i_1 i_2 \cdots i_{m-1} j_m i_{m+1} \cdots i_{n-1} i i_{n+1} \cdots i_N} A_{i j_n}.$$

Since

$$\kappa_{i_1 i_2 \cdots i_{m-1} j_m i_{m+1} \cdots i_{n-1} i i_{n+1} \cdots i_N} = \sum_{j=1}^{I_m} \chi_{i_1 i_2 \cdots i_{m-1} j i_{m+1} \cdots i_{n-1} i i_{n+1} \cdots i_N} B_{j j_m},$$

then

$$\omega_{i_1 i_2 \cdots i_{m-1} j_m i_{m+1} \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i=1}^{I_n} \sum_{j=1}^{I_m} \chi_{i_1 i_2 \cdots i_{m-1} j i_{m+1} \cdots i_{n-1} i i_{n+1} \cdots i_N} B_{j j_m} A_{i j_n}.$$

Based on the above analysis, we can obtain that $\gamma = \omega$. That is $\chi \times_n A \times_m B = \chi \times_m B \times_n A$.

Proof 37: By lemma 9, we can get that $\|A\|_F^2 = Trace(A^T A)$, $\|UA\|_F^2 = Trace((UA)^T UA)$. That is $\|UA\|_F^2 = Trace(A^T U^T UA)$. From the properties of orthogonal matrices, we can obtain that $U^T U = E$. Then $\|UA\|_F^2 = Trace(A^T A) = \|A\|_F^2$.

From the properties of Frobenius norm, we can get that $\|AU\|_F^2 = \|(AU)^T\|_F^2 = \|U^T A^T\|_F^2$. By lemma 9, $\|U^T A^T\|_F^2 = Trace((U^T A^T)^T U^T A^T)$. That is $\|U^T A^T\|_F^2 = Trace(AUU^T A^T)$. From the properties of orthogonal matrices, we can obtain that $UU^T = E$. Then $\|U^T A^T\|_F^2 = Trace(A^T A)$. Therefore, $\|AU\|_F^2 = \|A\|_F^2$.

Proof 38: From the definition of 2, we can get that $\|A\|_F^2 = \sum_{i=1}^{I} \sum_{j=1}^{J} A_{ij}^2$. Let $B = AU$, $B \in \mathbb{R}^{I \times J}$. $\|AU\|_F^2 = \|B\|_F^2 = \sum_{i=1}^{I} \sum_{j=1}^{J} B_{ij}^2$. Since $B_{ij} = \sum_{p=1}^{J} A_{ip} U_{pj}$, $\|AU\|_F^2 = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{p=1}^{J} A_{ip}^2 U_{pj}^2$. From the properties of diagonal matrices, we can get that $\|AU\|_F^2 = \sum_{i=1}^{I} \sum_{j=1}^{J} A_{ij}^2 U_{jj}^2$. Since $U_{jj}^2 \geqslant 0$, the minimum of $\sum_{i=1}^{I} \sum_{j=1}^{J} A_{ij}^2 U_{jj}^2$ is equivalent to the minimum of $\sum_{i=1}^{I} \sum_{j=1}^{J} A_{ij}^2$. Therefore, when $\|AU\|_F^2$ takes the minimum value, $\|A\|_F^2$ takes the minimum value.

Proof 39: From the known, we can get that $\phi(A) = \|AB\|_F^2 = \|AU \sum V\|_F^2$. Since V is a orthogonal matrix, by lemma 14, $\|AU \sum V\|_F^2 = \|AU \sum\|_F^2$. Let $C = AU$, $C \in \mathbb{R}^{I \times J}$. $\phi(A) = \|C \sum\|_F^2$. According to lemma 15, when $\|C \sum\|_F^2$ takes the minimum value, $\|C\|_F^2$ takes the minimum value. By lemma 14, we can obtain that $\|C\|_F^2 = \|AU\|_F^2 = \|A\|_F^2$. Since $\varphi(A) = \|A\|_F^2$, when $\phi(A)$ is minimum, $\varphi(A)$ is minimum.

Proof 40: Let $\lambda = \chi - v \prod_{n=1}^{N} \times_n C_n$, $\lambda \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. Then $F_{Pro1} = \|\lambda\|_F^2$. By lemma 8, we can get that $F_{Pro1} = \|\lambda\|_F^2 = \|\lambda_{(p)}\|_F^2$, $p \in [1, N]$. Let $A = \lambda_{(p)}$, $A \in \mathbb{R}^{(I_1 I_2 \cdots I_{p-1} I_{p+1} \cdots I_N) \times I_p}$. Then $F_{Pro1} = \|A\|_F^2$. According to lemma 11, let B satisfies the condition that $C_p B = E$, $B \in \mathbb{R}^{I_p \times J_p}$. then $B = V^{(p)} \Delta U^{(p)T}$, where $U^{(p)}$, Δ, $V^{(p)}$ can be obtained from the singular value decomposition of C_p, $C_p = U^{(p)} \sum^{(p)} V^{(p)T}$. $U^{(p)}$ and $V^{(p)}$ are orthogonal matrices, $\sum^{(p)}$ is the diagonal matrix. Δ is the inverse of the diagonal matrix $\sum^{(p)}$. Δ is the diagonal matrix.

From lemma 16, we can get that when $\|AB\|_F^2$ takes the minimum value, $\|A\|_F^2$ takes the minimum value. Then $F_{Pro1} = \|AB\|_F^2$. Since $\|AB\|_F^2 = \|\lambda_{(p)} B\|_F^2$, by lemma 10, we can get that $F_{Pro1} = \|\lambda_{(p)} B\|_F^2 = \|(\lambda \times_p B)_{(p)}\|_F^2$. $\lambda \times_p B = (\chi - v \prod_{n=1}^{N} \times_n C_n) \times_p B$. That is $\lambda \times_p B = \chi \times_p B - (v \prod_{n=1}^{N} \times_n C_n) \times_p B$. From lemma 13, we can obtain that $(v \prod_{n=1}^{N} \times_n C_n) \times_p B = (v \prod_{n=1, n \neq p}^{N} \times_n C_n) \times_p C_p \times_p B$.

From lemma 12, we can obtain that $(\nu \prod_{n=1,n\neq p}^{N} \times_n C_n)\times_p C_p \times_p B = (\nu \prod_{n=1,n\neq p}^{N} \times_n C_n)\times_p (C_p B)$. Since $C_p B = E$, then $(\nu \prod_{n=1}^{N} \times_n C_n)\times_p B = \nu \prod_{n=1,n\neq p}^{N} \times_n C_n$. Therefore, $\lambda \times_p B = \chi \times_p B - \nu \prod_{n=1,n\neq p}^{N} \times_n C_n$.

From the above analysis, we can get that $F_{Pro1} = \left\|(\lambda \times_p B)_{(p)}\right\|_F^2 = \left\|\lambda \times_p B\right\|_F^2$. That is $F_{Pro1} = \left\|\chi \times_p B - \nu \prod_{n=1,n\neq p}^{N} \times_n C_n\right\|_F^2$. According to the same principle, we can get the following formula by analogy.

$$F_{Pro1} = \left\|\chi \prod_{n=1}^{N} \times_n B_n - \nu\right\|_F^2 \tag{18}$$

B_n satisfies the following condition:

$$B_n = V^{(n)} \Delta^{(n)} U^{(n)T} \tag{19}$$

where $V^{(n)}$, $\Delta^{(n)}$ and $U^{(n)}$ can be obtained by performing singular value decomposition on matrix C_n. $C_n = U^{(n)} \sum^{(n)} V^{(n)T}$. $U^{(n)}$ and $V^{(n)}$ are orthogonal matrices. $\sum^{(n)}$ is a diagonal matrix. $\Delta^{(n)}$ is the inverse matrix of $\sum^{(n)}$.

SECOND APPENDIX

Proofs 41, 42 and 43 give the proof process of lemmas 18, 19 and 20, respectively.

Proof 41: Let $C = \nu_{(p)}$, $C \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$, $D = (\chi \prod_{n=1}^{N} \times_n B_n)_{(p)}$, $D \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$. Then $F_{sub1} = Trace(C^T D)$. Let $A = C^T D$, $A \in \mathbb{R}^{J_p \times J_p}$. From the matrix multiplication rule,, we can get that $A_{ij} = \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} C_{ik}^T D_{kj}$. According to definition 1, $F_{sub1} = Trace(A) = \sum_{j=1}^{J_p} A_{jj}$. Then $Trace(A) = \sum_{j=1}^{J_p} \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} C_{kj} D_{kj}$. When $k = i$, $\frac{\partial (C_{ij} D_{ij})}{\partial C_{ij}} = D_{ij}$. When $k \neq i$, $\frac{\partial (C_{kj} D_{kj})}{\partial C_{ij}} = 0$. Therefore, $\frac{\partial Trace(A)}{\partial C_{ij}} = D_{ij}$. That is $\frac{\partial F_{sub1}}{\partial C} = D$. $\frac{\partial F_{sub1}}{\partial \nu_{(p)}} = (\chi \prod_{n=1}^{N} \times_n B_n)_{(p)}$.

Proof 42: Let $C = \nu_{(p)}$, $C \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$, $D = (\chi \prod_{n=1}^{N} \times_n B_n)_{(p)}$, $D \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$. Then $F_{sub2} = Trace(D^T C)$. Let $A = D^T C$, $A \in \mathbb{R}^{J_p \times J_p}$. From the matrix multiplication rule,, we can get that $A_{ij} = \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} D_{ik}^T C_{kj}$. According to definition 1, $F_{sub2} = Trace(A) = \sum_{j=1}^{J_p} A_{jj}$. $F_{sub2} = \sum_{j=1}^{J_p} \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} D_{kj} C_{kj}$. When $k = i$, $\frac{\partial (D_{ij} C_{ij})}{\partial C_{ij}} = D_{ij}$. When $k \neq i$, $\frac{\partial (D_{kj} C_{kj})}{\partial C_{ij}} = 0$. Therefore, $\frac{\partial Trace(A)}{\partial C_{ij}} = D_{ij}$. $\frac{\partial F_{sub2}}{\partial C} = D$. $\frac{\partial F_{sub2}}{\partial \nu_{(p)}} = (\chi \prod_{n=1}^{N} \times_n B_n)_{(p)}$.

Proof 43: Let $C = \nu_{(p)}$, $C \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$, then $F_{sub3} = Trace(C^T C)$. Let $A = C^T C$, $A \in \mathbb{R}^{J_p \times J_p}$. According to definition 1, $F_{sub3} = Trace(A) = \sum_{j=1}^{J_p} A_{jj}$. From the matrix multiplication rule, we can get that $A_{ij} = \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} C_{ik}^T C_{kj}$. $Trace(A) = \sum_{j=1}^{J_p} \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} C_{jk}^T C_{kj}$. When $k = i$, $\frac{\partial (C_{ij} C_{ij})}{\partial C_{ij}} = 2C_{ij}$. When $k \neq i$, $\frac{\partial (C_{kj} C_{kj})}{\partial C_{ij}} = 0$. Therefore, $\frac{\partial Trace(A)}{\partial C_{ij}} = 2C_{ij}$. $\frac{\partial F_{sub3}}{\partial C} = 2C$. $\frac{\partial F_{sub3}}{\partial \nu_{(p)}} = \xi \nu_{(p)}$, where ξ is a constant, $\xi = 2$.

THIRD APPENDIX

Proofs 44, 45 and 46 give the proof process of lemmas 22, 23 and 24, respectively.

Proof 44: Let $A = \widehat{\chi}_{(p)}$, $A \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$, $D = \nu_{(p)}$, $D \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$, $G = A^T D$, $G \in \mathbb{R}^{J_p \times J_p}$. Then $F_{sub4} = Trace(G)$. According to definition 1, we can obtain that $Trace(G) = \sum_{j=1}^{J_p} G_{jj}$. Knowing from the rule of matrix multiplication, $G_{ij} = \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} A_{ik}^T D_{kj}$. Then $Trace(G) = \sum_{j=1}^{J_p} \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} A_{jk}^T D_{kj}$. That is $F_{sub4} = \sum_{j=1}^{J_p} \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} A_{kj} D_{kj}$. When $k = i$, $\frac{\partial (A_{ij} D_{ij})}{\partial A_{ij}} = D_{ij}$. When $k \neq i$, $\frac{\partial (A_{kj} D_{kj})}{\partial A_{ij}} = 0$. Therefore, $\frac{\partial F_{sub4}}{\partial A_{ij}} = D_{ij}$. $\frac{\partial F_{sub4}}{\partial \widehat{\chi}_{(p)}} = \nu_{(p)}$.

Proof 45: Let $A = \widehat{\chi}_{(p)}$, $A \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$, $D = \nu_{(p)}$, $D \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$, $G = D^T A$, $G \in \mathbb{R}^{J_p \times J_p}$. Then $F_{sub5} = Trace(G)$. According to definition 1, we can obtain that $Trace(G) = \sum_{j=1}^{J_p} G_{jj}$. Knowing from the rule of matrix multiplication, $G_{ij} = \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} D_{ik}^T A_{kj}$. Then $Trace(G) = \sum_{j=1}^{J_p} \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} D_{jk}^T A_{kj}$. That is $F_{sub5} = \sum_{j=1}^{J_p} \sum_{k=1}^{J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N} D_{kj} A_{kj}$. When $k = i$, $\frac{\partial (D_{ij} A_{ij})}{\partial A_{ij}} = D_{ij}$. When $k \neq i$, $\frac{\partial (D_{kj} A_{kj})}{\partial A_{ij}} = 0$. Therefore, $\frac{\partial F_{sub5}}{\partial A_{ij}} = D_{ij}$. $\frac{\partial F_{sub5}}{\partial \widehat{\chi}_{(p)}} = \nu_{(p)}$.

Proof 46: Let $A = \widehat{\chi}_{(p)}$, $A \in \mathbb{R}^{(J_1 J_2 \cdots J_{p-1} J_{p+1} \cdots J_N)\times J_p}$, $G = A^T A$, $G \in \mathbb{R}^{J_p \times J_p}$. Then $F_{sub6} = Trace(G)$. According to definition 1, we can obtain that

$Trace(G) = \sum_{j=1}^{J_p} G_{jj}$. Knowing from the rule of matrix multiplication, $G_{ij} = \sum_{k=1}^{J_1J_2\cdots J_{p-1}J_{p+1}\cdots J_N} A_{ik}^T A_{kj}$. Then $Trace(G) = \sum_{j=1}^{J_p}\sum_{k=1}^{J_1J_2\cdots J_{p-1}J_{p+1}\cdots J_N} A_{jk}^T A_{kj}$. That is $F_{sub6} = \sum_{j=1}^{J_p}\sum_{k=1}^{J_1J_2\cdots J_{p-1}J_{p+1}\cdots J_N} A_{kj}{}^2$. When $k = i$, $\frac{\partial(A_{ij}{}^2)}{\partial A_{ij}} = 2A_{ij}$. When $k \neq i$, $\frac{\partial(A_{ij}{}^2)}{\partial A_{ij}} = 0$. Therefore, $\frac{\partial F_{sub6}}{\partial A_{ij}} = 2A_{ij}$. $\frac{\partial F_{sub6}}{\partial \widehat{\chi}_{(p)}} = \xi\widehat{\chi}_{(p)}$, where ξ is a constant, $\xi = 2$.

FOURTH APPENDIX

Proofs 47, 48, 49, 50 and 51 give the proof process of lemmas 26, 27, 28, 29 and 30, respectively.

Proof 47: Let $D = AB$, $D \in \mathbb{R}^{P\times N}$, $G = DC$, $G \in \mathbb{R}^{P\times P}$. Then Trace(ABC) $= T$race(G). According to definition 1, we can get that Trace(G) $= \sum_{i=1}^{P} G_{ii}$. By the rule of matrix multiplication, $G_{ij} = \sum_{k=1}^{N} D_{ik}C_{kj}$. $D_{ij} = \sum_{q=1}^{M} A_{iq}B_{qj}$. Thus $G_{ij} = \sum_{k=1}^{N}\sum_{q=1}^{M} A_{iq}B_{qk}C_{kj}$. Then Trace(G) $= \sum_{i=1}^{P}\sum_{k=1}^{N}\sum_{q=1}^{M} A_{iq}B_{qk}C_{ki}$.

Let $Z = CA$, $Z \in \mathbb{R}^{N\times M}$, $S = ZB$, $S \in \mathbb{R}^{N\times N}$. Then Trace(CAB) $= T$race(S). According to definition 1, we can get that Trace(S) $= \sum_{k=1}^{N} S_{kk}$. By the rule of matrix multiplication, $Z_{mn} = \sum_{i=1}^{P} C_{mi}A_{in}$. $S_{mn} = \sum_{q=1}^{M} Z_{mq}B_{qn}$. That is $S_{mn} = \sum_{q=1}^{M}\sum_{i=1}^{P} C_{mi}A_{iq}B_{qn}$. Then Trace(S) $= \sum_{k=1}^{N}\sum_{q=1}^{M}\sum_{i=1}^{P} C_{ki}A_{iq}B_{qk}$. Therefore, Trace(S) $= T$race(G). Trace(ABC) $= T$race(CAB).

Proof 48: Let $A = \varphi(\widehat{\chi}^{(m)})$, $A \in \mathbb{R}^1$, $B = \widehat{\chi}_{vec}^{(m)}$, $B \in \mathbb{R}^{J_1J_2\cdots J_N}$. Then $f_\varphi = Trace(A^TB^T\mu) = ATrace(B^T\mu)$. Let $D = B^T\mu$, $D \in \mathbb{R}^1$. By the rule of matrix multiplication, $D_{ij} = \sum_{k=1}^{J_1J_2\cdots J_N} B_{ik}^T\mu_{kj} = \sum_{k=1}^{J_1J_2\cdots J_N} B_{ki}\mu_{kj}$. According to definition 1, $Trace(D) = \sum_{p=1}^{1} D_{pp}$. That is $Trace(D) = \sum_{p=1}^{1}\sum_{k=1}^{J_1J_2\cdots J_N} B_{kp}\mu_{kp}$. When $k = i$, $p = j$, $\frac{\partial(B_{ij}\mu_{ij})}{\partial B_{ij}} = \mu_{ij}$. Otherwise, $\frac{\partial(B_{kp}\mu_{kp})}{\partial B_{ij}} = 0$. Then $\frac{\partial Trace(D)}{\partial B} = \mu$. That is $\frac{\partial f_\varphi}{\partial \widehat{\chi}_{vec}^{(m)}} = \varphi(\widehat{\chi}^{(m)})\mu$.

Proof 49: Let $A = \varphi(\widehat{\chi}^{(m)})$, $A \in \mathbb{R}^1$, $B = \widehat{\chi}_{vec}^{(m)}$, $B \in \mathbb{R}^{J_1J_2\cdots J_N}$. Then $f_\chi = Trace(\mu^TBA) = ATrace(\mu^TB)$. Let $D = \mu^TB$, $D \in \mathbb{R}^1$. According to definition 1, $Trace(D) = \sum_{p=1}^{1} D_{pp}$. By the rule of matrix multiplication, $D_{ij} = \sum_{k=1}^{J_1J_2\cdots J_N} \mu_{ik}^T B_{kj} = \sum_{k=1}^{J_1J_2\cdots J_N} \mu_{ki}B_{kj}$. Then $Trace(D) = \sum_{p=1}^{1}\sum_{k=1}^{J_1J_2\cdots J_N} \mu_{kp}B_{kp}$. When $k = i$, $p = j$, $\frac{\partial(\mu_{ij}B_{ij})}{\partial B_{ij}} = \mu_{ij}$. Otherwise, $\frac{\partial(\mu_{kp}B_{kp})}{\partial B_{ij}} = 0$. Then $\frac{\partial Trace(D)}{\partial B} = \mu$. That is $\frac{\partial f_\chi}{\partial \widehat{\chi}_{vec}^{(m)}} = \varphi(\widehat{\chi}^{(m)})\mu$.

Proof 50: Let $B = \widehat{\chi}_{vec}^{(m)}$, $B \in \mathbb{R}^{J_1J_2\cdots J_N}$. Then $f_T = Trace(\mu^TBB^T\mu)$. According to lemma 26, $Trace(\mu^TBB^T\mu) = Trace(B^T\mu\mu^TB)$. Let $D = \mu\mu^T$, $D \in \mathbb{R}^{(J_1J_2\cdots J_N)\times(J_1J_2\cdots J_N)}$. $f_T = Trace(B^TDB)$. Let $H = B^TD$, $H \in \mathbb{R}^{1\times(J_1J_2\cdots J_N)}$, $G = HB$, $G \in \mathbb{R}^1$. According to definition 1, $Trace(G) = \sum_{p=1}^{1} G_{pp}$. By the rule of matrix multiplication, $G_{ij} = \sum_{k=1}^{J_1J_2\cdots J_N} H_{ik}B_{kj}$. $H_{mn} = \sum_{q=1}^{J_1J_2\cdots J_N} B_{mq}^T D_{qn} = \sum_{q=1}^{J_1J_2\cdots J_N} B_{qm}D_{qn}$. Then $G_{ij} = \sum_{k=1}^{J_1J_2\cdots J_N}\sum_{q=1}^{J_1J_2\cdots J_N} B_{qi}D_{qk}B_{kj}$. $Trace(G) = \sum_{p=1}^{1}\sum_{k=1}^{J_1J_2\cdots J_N}\sum_{q=1}^{J_1J_2\cdots J_N} B_{qp}D_{qk}B_{kp}$. When $q = i$, $p = j$, $\frac{\partial(B_{ij}D_{ik}B_{kj})}{\partial B_{ij}} = D_{ik}B_{kj}$. When $k = i, p = j$, $\frac{\partial(B_{qj}D_{qi}B_{ij})}{\partial B_{ij}} = B_{qj}D_{qi} = D_{iq}^TB_{qj}$. Otherwise, $\frac{\partial(B_{qp}D_{qk}B_{kp})}{\partial B_{ij}} = 0$. Therefore, $\frac{\partial Trace(G)}{\partial B} = DB + D^TB$. Since $D = D^T$, $\frac{\partial Trace(G)}{\partial B} = 2DB$. That is $\frac{\partial f_T}{\partial \widehat{\chi}_{vec}^{(m)}} = \xi\mu\mu^T\widehat{\chi}_{vec}^{(m)}$, where ξ is a constant, $\xi = 2$.

Proof 51: Let $A = C_n$, $A \in \mathbb{R}^{J_n\times I_n}$. Then $F_C = Trace(A^TA)$. Let $D = A^TA$, $D \in \mathbb{R}^{I_n\times I_n}$. According to definition 1, $Trace(D) = \sum_{i=1}^{I_n} D_{ii}$. By the rule of matrix multiplication, $D_{ij} = \sum_{k=1}^{J_n} A_{ik}^TA_{kj} = \sum_{k=1}^{J_n} A_{ki}A_{kj}$. Then $Trace(D) = \sum_{i=1}^{I_n}\sum_{k=1}^{J_n} A_{ki}^2$. When $k = m$, $i = n$, $\frac{\partial(A_{mn}^2)}{\partial A_{mn}} = 2A_{mn}$. Otherwise, $\frac{\partial(A_{ki}^2)}{\partial A_{mn}} = 0$. Therefore, $\frac{\partial Trace(D)}{\partial A} = 2A$. That is $\frac{\partial F_C}{\partial C_n} = \xi C_n$, where ξ is a constant, $\xi = 2$.

REFERENCES

[1] L. A. D. S. Gruginskie and G. L. R. Vaccaro, "Lawsuit lead time prediction: Comparison of data mining techniques based on categorical response variable," *PLoS ONE*, vol. 13, no. 6, Jun. 2018, Art. no. e0198122.
[2] G. W. Manes and E. Downing, "Overview of licensing and legal issues for digital forensic investigators," *IEEE Secur. Privacy Mag.*, vol. 7, no. 2, pp. 45–48, Mar. 2009.
[3] L. Jing, C. Shen, L. Yang, J. Yu, and M. K. Ng, "Multi-label classification by semi-supervised singular value decomposition," *IEEE Trans. Image Process.*, vol. 26, no. 10, pp. 4612–4625, Oct. 2017.
[4] W. Li, Q. Jing, Z. Yu, and D. Li, "A social recommendation method based on trust propagation and singular value decomposition," *J. Intell. Fuzzy Syst.*, vol. 32, no. 1, pp. 1–10, 2016.
[5] K. Wimalawarne, R. Tomioka, and M. Sugiyama, "Theoretical and experimental analyses of tensor-based regression and classification," *Neural Comput.*, vol. 28, no. 4, pp. 686–715, Apr. 2016.
[6] Y.-H. Taguchi, "Correction: Tensor decomposition-based unsupervised feature extraction applied to matrix products for multi-view data processing," *PLoS ONE*, vol. 13, no. 7, Jul. 2018, Art. no. e0200451.
[7] X. Zheng, W. Ding, Z. Lin, and C. Chen, "Topic tensor factorization for recommender system," *Inf. Sci.*, vol. 372, pp. 276–293, Dec. 2016.

[8] P. Kim, "Convolutional neural network," in *MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence*. Berkeley, CA, USA: Apress, 2017, pp. 121–147, doi: 10.1007/978-1-4842-2845-6_6.
[9] A. Blanco, M. Delgado, and M. C. Pegalajar, "A genetic algorithm to obtain the optimal recurrent neural network," *Int. J. Approx. Reasoning*, vol. 23, no. 1, pp. 67–83, Jan. 2000.
[10] W. Yin, H. Schätze, B. Xiang, and B. Zhou, "ABCNN: Attention-based convolutional neural network for modeling sentence pairs," *Trans. Assoc. Comput. Linguistics*, vol. 4, pp. 259–272, Dec. 2016.
[11] T. Liu, S. Yu, B. Xu, and H. Yin, "Recurrent networks with attention and convolutional networks for sentence representation and classification," *Int. J. Speech Technol.*, vol. 48, no. 10, pp. 3797–3806, Oct. 2018.
[12] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Comput.*, vol. 12, no. 10, pp. 2451–2471, Oct. 2000.
[13] Y. Yao and Z. Huang, "Bi-directional LSTM recurrent neural network for Chinese word segmentation," in *Neural Information Processing*, H. Akira, S. Ozawa, K. Doya, K. Ikeda, M. Lee, and D. Liu, Eds. Cham, Switzerland: Springer, 2016, pp. 345–353.
[14] G. Molinar, N. Popovic, and W. Stork, "From data points to ampacity forecasting: Gated recurrent unit networks," in *Proc. IEEE 4th Int. Conf. Big Data Comput. Service Appl. (BigDataService)*, Mar. 2018, pp. 200–207.
[15] L. Zhang, Y. Zhou, X. Duan, and R. Chen, "A hierarchical multi-input and output bi-gru model for sentiment analysis on customer reviews," *Iop Conf.*, vol. 322, Oct. 2018, Art. no. 062007.
[16] H. Späth, "Algorithm 39 clusterwise linear regression," *Computing*, vol. 22, no. 4, pp. 367–373, Dec. 1979.
[17] E. Blais, R. O'Donnell, and K. Wimmer, "Polynomial regression under arbitrary product distributions," *Mach. Learn.*, vol. 80, nos. 2–3, pp. 273–294, Sep. 2010.
[18] G. C. Cawley and N. L. C. Talbot, "Reduced rank kernel ridge regression," *Neural Process. Lett.*, vol. 16, no. 3, pp. 293–302.
[19] X. Yang and W. Wushao, "Ridge and lasso regression models for cross-version defect prediction," *IEEE Trans. Rel.*, vol. 67, no. 3, pp. 885–896, Sep. 2018.
[20] C. De Mol, E. De Vito, and L. Rosasco, "Elastic-net regularization in learning theory," *J. Complex.*, vol. 25, no. 2, pp. 201–230, Apr. 2009.

XIAODING GUO (Member, IEEE) received the B.S. degree in computer science from Xidian University, Xi'an, China, in 2015. She is currently pursuing the Ph.D. degree in computer science with the Harbin Institute of Technology. Her research interests include big data and data mining.

HONGLI ZHANG received the B.S. degree in computer science from Sichuan University, Chengdu, China, in 1994, and the Ph.D. degree in computer science from the Harbin Institute of Technology (HIT), Harbin, China, in 1999. In 2012, she was a Visiting Scholar with North Carolina State University, Raleigh, NC, USA. She is currently a Professor with the School of Computer Science and Technology, HIT, and the Vice Director of the National Computer Information Content Security Key Laboratory. Her research interests include network and information security, network measurement and modeling, and parallel processing.

LIN YE received the Ph.D. degree from the Harbin Institute of Technology, Harbin, China, in 2011. From January 2016 to January 2017, he was a Visiting Scholar with the Department of Computer and Information Sciences, Temple University, USA. His current research interests include network security, peer-to-peer networks, network measurement, and cloud computing.

SHANG LI (Graduate Student Member, IEEE) received the B.S. and M.S. degrees in computer science from the Harbin Institute of Technology (HIT), Harbin, China, in 2012 and 2014, respectively, where he is currently pursuing the Ph.D. degree in computer science. His research interests include legal intelligence, data mining, and social network analysis.

GUANGYAO ZHANG received the M.S. degree in computer application technology from Dalian Maritime University, Dalian, China, in 2003. He is currently the Director of the Heilongjiang Branch, CNCERT.