Fuzzy Rule-Based Models of High-Dimensional Systems: Design and Analysis

by

Hanyu E

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering
University of Alberta

Abstract

Fuzzy models, especially fuzzy rule-based models, have received substantial attention as an important pursuit in the design and analysis of intelligent systems. In rule-based architectures, functional (Takagi-Sugeno) fuzzy rule-based models have been studied intensively resulting in a wealth of design strategies and applications. Due to the concentration effect, high dimensional data pose a genuine challenge to the efficient development process, quality of the models, and their ensuing interpretability. This research addresses these challenges in several fundamental and original ways.

First, an original relational factorization based on fuzzy relational calculus and engaging triangular norms and conorms is developed. In terms of the enhancement of models, we propose two schemes. One is to construct a fuzzy rule-based model based on fuzzy relational factorization. The factorization facilitates an interpretable, logic-oriented encoding of conditions of the rules. Thus, the clustering algorithm commonly used in the design of TS models is replaced by the factorization so that the concentration effect is avoided. The other is to introduce and analyze the concept of distributed fuzzy rule-based models. The mechanisms of ensemble learning and gradient boosting are explored to construct the rules. In their realization, the data are sampled randomly and applied to each distributed rule-based model. Then the gradient boosting algorithm is involved to improve the quality of the models. Subsequently, some essential refinements of the fuzzy clustering method used in the formation of conditions of the rules are developed to further enhance the performance of the rule-based models. The augmentations of the clustering algorithm include: (i) an accommodation of extreme (minimal and maximal) values encountered in the output variable, (ii) a reduction of redundancy of the rules which is the result of overlap

existing among fuzzy sets forming the condition parts of the rules. (iii) the development of the core (granular) structure of rules and analysis of their features. The proposed methods, novel identification of the major characteristics of data, and a novel construction of the model, constitute the originality of this thesis. The experimental results obtained for publicly available data are reported along with a solid comparative analysis.

Preface

This research is completed by Hanyu E under the guidance of Prof. Witold Pedrycz, which is supported by Grad Research Assistantship Fellowship (GRAF), Faculty of Engineering and Future Energy Systems (FES), Natural Sciences and Engineering Research Council of Canada (NSERC).

Chapter 1 introduces the overall structure of the thesis and describes the motivation, objectives, and originality of the thesis.

Chapter 2 reviews the basic algorithms used in this article, as well as some background knowledge to facilitate readers to understand the subsequent content.

Chapter 3 contains the material published as H. E, Y. Cui, W. Pedrycz, and Z. Li, "Fuzzy Relational Matrix Factorization and Its Granular Characterization in Data Description," *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 3, pp. 794-804, Dec. 2020. I was responsible for experiment development, data collection and analysis, and manuscript composition. Y. Cui focused on data collection and analysis. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition. Z. Li was involved with manuscript composition.

Chapter 4 contains the material published as H. E, Y. Cui, W. Pedrycz, A. R. Fayek, Z. Li, and J. Li, "Design of Fuzzy Rule-Based Models with Fuzzy Relational Factorization", *Expert systems with applications*, vol. 206, 117904, Nov. 2022. I was responsible for experiment development, data collection and analysis, and manuscript composition. Y. Cui focused on data collection and analysis. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition. A. R. Fayek and Z. Li

were involved with the manuscript composition. J. Li was involved with the experiment development.

Chapter 5 contains the material submitted to *IEEE Transactions on Fuzzy Systems* by H. E, Y. Cui, W. Pedrycz, and Z. Li, "Design of distributed rule-based models in the presence of large data." I was responsible for experiment development, data collection and analysis, and manuscript composition. Y. Cui focused on data collection and analysis. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition. Z. Li was involved with manuscript composition.

Chapter 6 contains the material published by H. E, Y. Cui, W. Pedrycz, and Z. Li, "Enhancements of rule-based models through refinements of Fuzzy C-Means," *Knowledge-Based Systems*, vol. 170, pp. 43-60, Apr. 2019. I was responsible for experiment development, data collection and analysis, and manuscript composition. Y. Cui focused on data collection and analysis. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition. Z. Li was involved with manuscript composition.

Acknowledgements

I would like to thank my supervisor Professor Witold Pedrycz, for being his student is the pride of my life. It was he who lead me to the path of scientific research. From Professor W. Pedrycz, I have not only learned knowledge and ideas, but also a serious and responsible attitude, a desire for knowledge and perseverance. These will be the precious wealth in my future life and work.

I would like to express my gratitude to all the professors who gave me valuable comments and suggestions. Which are Prof. Petr Musilek, Prof. Marek Reformat, Prof. Irene Cheng, Prof. Jie Han, Prof. Witold Krzymien, Prof. Emil M. Petriu, Prof. Aminah Robinson Fayek, Prof. Ergun kuru.

I would like to thank my teachers and friends who have helped me at every important point in my study journey. It is their dedication and companionship that made me what I am today. Especially, Zhiwu Li and Changcun Kan have provided tremendous help in my undergraduate and high school study respectively, guiding me like a beacon in the night sky.

I am grateful to all my fellow lab mates and friends encountered at the University of Alberta and Edmonton; with them, I felt at home in Edmonton and fell in love with the city.

Finally, I would like to thank my parents, Baosheng E and Qingzhi Han, who raised me, educated me, gave me the best living and learning environment, and worked hard for me. Their love for me will always be the driving force for me to move forward.

Hanyu E

University of Alberta

June 2022

Table of Contents

List of Tables

List of Figures

List of Symbols

X	Data input matrix
$\hat{X}$	Re-constructed data matrix
$target$	Data target
$\hat{y}$	Model output
N	Number of data
n	Data dimensionality
c	Number of clusters
Q	Performance index
$\bar{Q}$	Average of performance index
$RMSE$	Root Mean Squared Error
MSE	Mean Squared Error
sp	Specificity
cov	Coverage
W	Weight matrix

List of Abbreviations

FCM	Fuzzy C-Means
TS model	Takagi-Sugeno model
NMF	Non-negative matrix factorization
LSE	Least square error
ADAM	Adaptive moment estimation

Chapter 1 Introduction

In the late seventies, the idea of expert systems was proposed, mainly realized as a collection of 'if-then' rules. They were used to solve complex problems by identifying and describing rules [1]. Rules are characterized by modularity and were originally studied as a way of knowledge representation [2], [3]. Subsequently, fuzzy rule-based models (TS models) [4] were proposed by T. Takagi and M. Sugeno in 1985, which are constructed by if-then rules based on the fuzzy system [5]–[19].

Nowadays, we have entered the era of big data [20]–[24], in which data storage and data processing have become demanding [25]–[29]. As early as 1966, Bellman introduced the concept of the curse of dimensionality when considering problems in dynamic programming [30]. Subsequently, this concept is used to describe issues that arise when analyzing and organizing data in high-dimensional space, such as combinatorial explosion, concentration effect, etc. To deal with high-dimensional problems, two approaches are often used. The most intuitive way is to analyze and extract information from the data and discard the redundant one, thus achieving dimensionality reduction There are numerous approaches and architectures, say autoencoders, non-negative matrix factorization, principal component analysis (PCA), independent component analysis (ICA), random projection, among others. The second approach is to optimize and enhance the traditional models when coping with big data. We highlight the two main avenues viz. the realization of the process of dimensionality reduction and model optimization in Fig. 1.1.

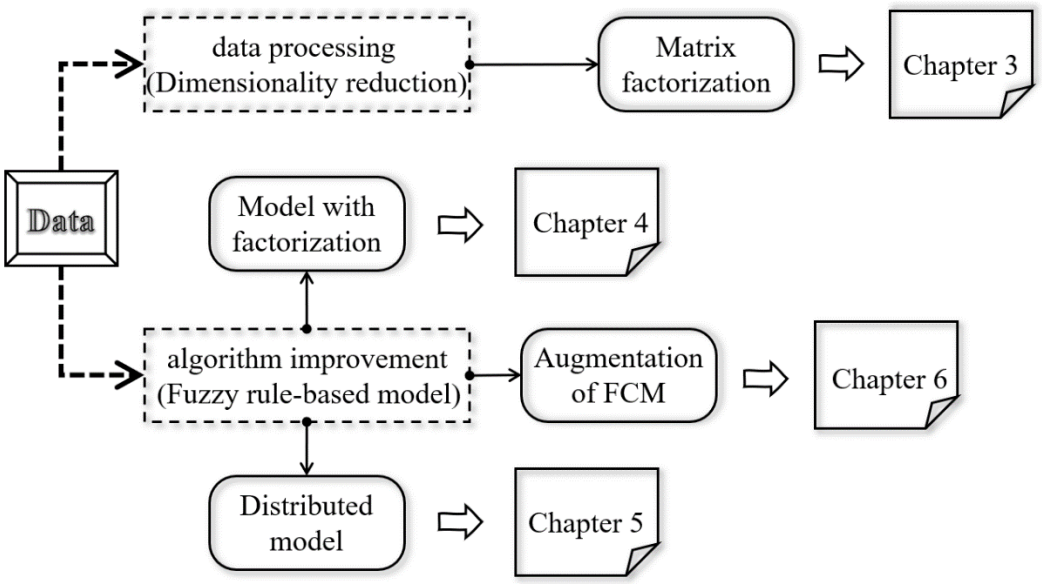


Fig. 1.1. Outline of the thesis.

To deal with high-dimensional problems, it is evident that dimensionality reduction is the most direct way. Thus, we propose a method to decrease dimensionality, which is the fuzzy relational matrix factorization in Chapter 3. In addition, high-dimensional problems can be solved by several improvement methods. A novel fuzzy rule-based model based on matrix factorization is developed in Chapter 4. The advanced distributed fuzzy rule-based model is proposed in Chapter 5. In Chapter 6, three different methods for the augmentation of the Fuzzy *C*-Means (FCM) are presented.

1.1 Motivation

With the information explosion, rule-based models are unavoidably exposed to high dimensional and large volumes of data. Usually, FCM plays a vital role in the construction of the condition parts of rules in rule-based models. However, FCM performs poorly when clustering high-dimensional data, which in the sequel leads to the poor performance of fuzzy rule-based

models. To address this issue, we consider two ways, namely dimensionality reduction and enhancement of the model. There are many methods proposed for dealing with high-dimensional data [31]–[33] However, information loss being the result of dimensionality reduction is inevitable. Thus, it becomes urgent to develop a new method that can retain more information. On the other hand, several argumentation methods on the rule-based models are also considered.

1.2 Objectives and Originality

The key objectives are as follows.

- The concept of relational decomposition of high dimensional data

We introduce triangular norms and triangular conorms and develop a matrix factorization model based on them. By involving those concepts, a learning scheme of three methods: single layer, two-level, and two-relation factorization are designed.

- The rule-based model enhanced by logic-based processing

We design models by capturing the underlying relationships present in the input-output data. Through the intermediate product of the matrix decomposition process, we reveal the relation between the data and the model and apply them to the fuzzy rule-based model.

- The concept of the distributed model

We develop a comprehensive design of an ensemble of rule-based models, producing a distributed architecture to cope with the high-dimensional data. The rule-based model is used in distributed models to deal with the data with chosen features. Then, a gradient boosting algorithm is applied to train the model to improve the model results.

- The improvement methods for the clustering algorithm

Three methods are developed: the first one is to make sure all values are covered by adjusting the prototype. The second one is to reduce spurious activation levels of rules. The third is targeted at rules, where granular rules instead of the original (numeric) ones are proposed.

The originality of the research in this study exhibits several facets.

- We develop a fuzzy relational matrix factorization based on the combination of *t*-norms and *t*-conorms, which is a new matrix decomposition method. By Introducing the logic-based triangular norms and conorms, we improve the performance and interpretability of the results of decomposition.
- A new idea is presented that the high dimensional data are decomposed to their low-dimensional counterparts and some optimized transformation relations, which can be regarded as a collection of logic expressions (*or-and* or *and-or* composition), linking low dimensional data with the original data via weighted disjunctive or conjunctive logic formulas.
- A novel distributed architecture based on ensemble learning is proposed, and a gradient boosting algorithm is used to improve the performance of the model.

1.3 Organization

The thesis is structured into the following chapters. In order to facilitate the understanding of the research, Chapter 2 reviews a number of documented methods such as fuzzy clustering, fuzzy rule-based model, matrix decomposition, ensemble learning, gradient descent algorithm, evaluation criteria, etc.

Chapter 3 touches upon a matrix factorization scheme based on triangular fuzzy norms. Then, the model structure of the two-layer matrix decomposition is presented. Finally, the idea of a matrix decomposition model based on granularity is developed.

In Chapter 4, we are concerned with the encoding mechanisms realized with the aid of intervals and fuzzy sets. Then we introduce a rule-based model augmented based on fuzzy factorization, which has better relative performance than traditional models.

Proposed in Chapter 5 is a distributed rule-based model based on the idea of Ensemble Learning, which combines with the gradient boosting algorithm for further optimization.

In Chapter 6, three enhancement mechanisms based on the core level of the traditional TS model are proposed, which are realized by introducing extreme values of input data, reducing unnecessary activation levels, and introducing the concept of information granule to enhance the performance of the traditional TS model.

In Chapter 7, we draw the main conclusions of this research and identify several promising directions for future research.

Chapter 2 Literature Review

In this chapter, we concisely review the algorithms and methods used throughout this study, including clustering algorithm, fuzzy rule-based model, matrix factorization, ensemble learning, optimization algorithm, cross validation, performance indices, and information granules.

2.1 Fuzzy clustering

Clustering is a process of grouping objects into several clusters by minimizing the distance between the data and the cluster centers. Clustering contains two types, clustering and soft clustering (fuzzy clustering). In the former, each data can only be assigned to one cluster, while for fuzzy clustering, an object may belong to multiple clusters with the corresponding degrees of belongingness (membership grades).

Fuzzy *C*-Means (FCM) is a commonly used fuzzy clustering method. FCM was firstly introduced by J.C. Dunn [34], which was further improved by J.C. Bezdek [35]. The aim of FCM is to find the clustering centers (called prototypes) and partition matrix (used to describe the degree of membership of data to the clusters) which is through an iterative process. The objective function is the sum of weighted distance between data $X=\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$ in R^n-dimensional space and c prototypes $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_c$, see equation (2.1).

$$Q = \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik}^{m} \|\boldsymbol{x}_k - \boldsymbol{v}_i\|^2 \tag{2.1}$$

where the fuzzification coefficient m is greater than 1. u_{ik} is the component of partition matrix, standing for the membership grade of the k^{th} data to the i^{th} cluster. The sum of clustering the k^{th}

data into c clusters is 1, i.e., $\sum_{i=1}^{c} u_{ik} = 1$, and $u_{ik} \in [0, 1]$. $\boldsymbol{x}_k$ is the k^{th} data point, $\boldsymbol{v}_i$ is the i^{th} prototype, and the distance between them is

$$\|\boldsymbol{x}_k - \boldsymbol{v}_i\|^2 = \sum_{j=1}^{n} \frac{\left(x_{kj} - v_{ij}\right)^2}{\sigma_j^2} \tag{2.2}$$

where σ_j stands for deviation of the j^{th} attribute of X.

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|\boldsymbol{x}_k - \boldsymbol{v}_i\|}{\|\boldsymbol{x}_k - \boldsymbol{v}_j\|} \right)^{2/(m-1)}} \tag{2.3}$$

$$\boldsymbol{v}_i = \frac{\sum_{k=1}^{N} u_{ik}^m \boldsymbol{x}_k}{\sum_{k=1}^{N} u_{ik}^m} \tag{2.4}$$

To achieve the minimum of Q in (2.1), the elements in the partition matrix and prototypes are updated in an iterative way.

The detailed implementation of FCM is shown as follows:

Input: data X, the number of clusters c, fuzzification coefficient m, threshold τ, maximal iteration max_iter

Output: partition matrix and cluster centers

Randomly initialize cluster centers $\boldsymbol{v}_i$, $i = 1,2, \ldots c$, $iter = 0$.

Compute element of partition matrix u_{ik}, $k = 1,2, \ldots N$, value of objective function Q

while $Q > \tau$

    Compute u_{ik}, $\boldsymbol{v}_i$ by (2.3) and (2.4).

    Compute Q by (2.1).

    $iter = iter + 1$

if $iter > max\_iter$

break

end

end

Return partition matrix and cluster centers

The objective function (2.1) is influenced by the two main parameters c and m. It is evident that when the number of clusters c is high enough, the centers are close to or even equal to each data. Fig. 2.1 shows the membership functions as a function of m, where m is 1.05, 2, and 4. It is apparent that with the increase of m, the shape of the partition matrix becomes steep. When m is close to 1, the partition matrix tends to be crisp.

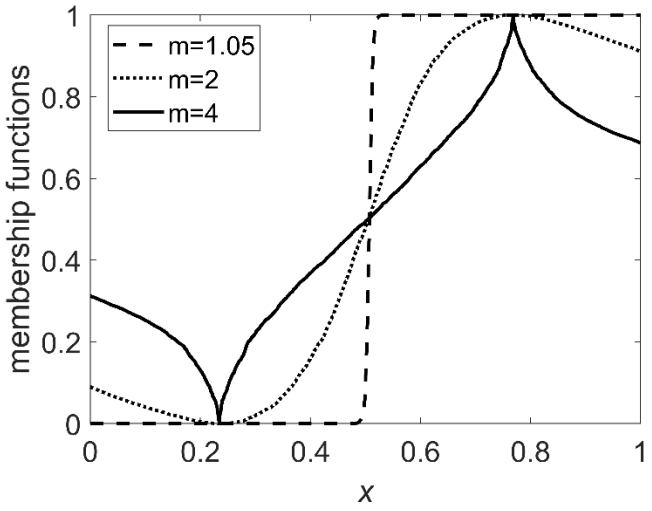


Fig. 2.1. Membership functions as a function of m.

There have been some related studies on the development of fuzzy clustering. Alata optimized the fuzzy coefficients by introducing the GA algorithm to improve the performance of the FCM algorithm [36]. In addition, the FCM was optimized with the introduction of gradient descent algorithm [37]. Pedrycz proposed the idea of collaborative fuzzy clustering, which aims at exploring the data structure [38], [39]. Suh proposed a two-level fuzzy clustering method that

involves adaptively expanding and merging convex polytopes, making it possible to represent an arbitrarily distributed data set [40]. Timme proposed to optimize the traditional clustering algorithm by introducing a mutual repulsion of the clusters [41]. At present, fuzzy clustering has been widely used in many fields, such as data mining [42], [43], pattern recognition [44], [45], and image analysis [46], [47].

It is undeniable that the fuzzy clustering problem under high-dimensional data has been a hot topic. For example, Mei developed a new fuzzy clustering method by dividing the ensemble, which performs better with high-dimensional data problems [48]. Eschrich proposed to deal with the high-dimensional clustering problem by reducing the number of distinct patterns [49]. A combination of multi-objective evolutionary algorithm and fuzzy clustering algorithm was proposed, which has high-quality clustering when faced with high dimensional data [50]. Nevertheless, clustering realized on high-dimensional data is still a topic worthy of further study. Chang proposed a novel fuzzy c-means (FCM) model with sparse regularization to deal with high-dimensional data [51]. A fuzzy clustering algorithm based on Hsim was proposed, which is used as the similarity measure of high dimensional data [52].

2.2 Fuzzy rule-based models

Fuzzy rule-based models are one of the essential application areas of fuzzy sets [53] and fuzzy logic [54]–[65]. By formulating *if-then* rules over the input and output space [4], [5] [66], [67], fuzzy rule-based models use fuzzy sets to describe and handle complex nonlinear relationships, where '*if*' is the condition part, and '*then*' is the conclusion part. Assume that we have available input data $\boldsymbol{x}_k, k = 1, 2, \ldots, N$, which is regarded as an n-dimensional vector in R^n. The structure of the rules is

$$-\text{if } \boldsymbol{x} \text{ is } A_i(\boldsymbol{x}), \text{then } y \text{ is } f_i(\boldsymbol{x}) \tag{2.5}$$

where $i = 1,2,\dots,c$, c is the number of fuzzy rules. In the condition part, A_i is a fuzzy set (information granule) while $A_i(\boldsymbol{x})$ denotes the degree of $\boldsymbol{x}$ belonging to i-th rule. In the conclusion part, $f_i(\boldsymbol{x})$ describes the local behavior of the model. It is usually a constant, linear or polynomial function. Where constant ($f_i(\boldsymbol{x}) = w_i$) and linear function ($f_i(\boldsymbol{x}) = a_i^0 + a_i^1 x_1 + \cdots + a_i^n x_n$) are widely studied due to their efficiency and sound performance. The model output is taken as the weighted aggregation of c outputs of the rules, namely

$$\hat{y} = \sum_{i=1}^{c} A_i(\boldsymbol{x}) f_i(\boldsymbol{x}) \Big/ \sum_{i=1}^{c} A_i(\boldsymbol{x}) \tag{2.6}$$

Due to the property of the partition matrix, $\sum_{i=1}^{c} A_i(\boldsymbol{x}) = 1$, the output is expressed as

$$\hat{y} = \sum_{i=1}^{c} A_i(\boldsymbol{x}) f_i(\boldsymbol{x}) \tag{2.7}$$

Design of condition and conclusion parts of rules

Start from condition part, membership functions are generated by clustering algorithm. The clustering algorithm could be realized based on the input data *X* or a collection of input-output pairs $(\boldsymbol{x}_k, target_k)$, $k = 1,2,\dots,N$. Thus, the prototypes will be $\boldsymbol{v}_i$ positioned in input space and w_i in output space, $i = 1,2,\dots,c$. When it comes to the conclusion part, first we need to define the structure of the local function. Let us consider constant function, the rule could be written as

$$-\text{if } \boldsymbol{x} \text{ is } A_i, \text{then } y \text{ is } w_i \tag{2.8}$$

It is a concise way, as mentioned, w_i comes from the clustering algorithm. Then we introduce two well-known linear functions,

$$-\text{if } \boldsymbol{x} \text{ is } A_i, \text{then } y \text{ is } a_i^0 + a_i^1 x_1 + \cdots + a_i^n x_n \tag{2.9}$$

where a_i^0 is the i^{th} constant value, and $\boldsymbol{a}_i = [a_i^1, a_i^2, \cdots, a_i^n]$ is the consequent parameter that need to be optimized in the i^{th} rule [68].

$$-\text{if } \boldsymbol{x} \text{ is } A_i, \text{then } y \text{ is } w_i + \boldsymbol{a}_i^T(\boldsymbol{x} - \boldsymbol{v}_i) \tag{2.10}$$

The estimation of parameters $\boldsymbol{a}_i$ and a_i^0 is realized based on Least Square Error (LSE). The objective function is to minimize the distance between model output $\hat{y}$ and the corresponding target.

At present, fuzzy rule-based models have been developed in several ways. Hu designed the bagging and boosting mechanisms for assembling fuzzy rule-based models and demonstrated that the performance of the ensemble model was superior to the traditional single model for most datasets [69]. The distributed way and hierarchically driven way of rule development were discussed in [70]. Kacimi improved fuzzy models by optimizing simultaneously the membership functions, the scaling factor parameters, and the fuzzy rule conclusions with a mixed-coding particle swarm optimization algorithm (PSO) [71]. Mamaghani was concerned with the

development of fuzzy models realized with the aid of genetic programming (GP). The proposed architecture employs GP to form fuzzy logic expressions involving logic operators and information granules (fuzzy sets) located in the input space [72]. A methodology for the encoding of the chromosome of a genetic algorithm (GA) was described in [73]. The encoding procedure is applied to the problem of automatically generating fuzzy rule-based models from data. Precup suggested a structure for prosthetic hand myoelectric-based control systems and a set of evolving Takagi-Sugeno-Kang (TSK) fuzzy models to characterize the finger dynamics of the human hand for the myoelectric control of prosthetic hands mathematically [74].

However, fuzzy rule-based models still face some challenges, especially in the context of the high dimensional data. High dimensional data has a negative effect on the accuracy of the clustering algorithm, which is an important tool to construct the condition part of rule-based models. Thus, the reliability of rule-based models will also be influenced negatively.

2.3 Non-negative matrix factorization

Non-negative matrix factorization (NMF) is one of the well-known methods of dimensionality reduction. The difference between NMF and principal component analysis (PCA) is that the PCA components are orthogonal to each other, while the NMF components are all non-negative and therefore construct a non-orthogonal basis [75], [76]. Non-negative matrix factorization is an algorithm in multivariate analysis and stemming from linear algebra, where the original data X are factorized into two matrices: basis matrix H and weight matrix R, with the property that all three matrices have no negative elements. The dimensionality of the matrix H is much smaller than the original dataset. In this process, we can obtain two smaller matrices by decomposing a larger matrix, where the matrix H represents the data after the dimensionality

reduction, and R is the weight matrix, which represents the weight of variables in the original data. In other words, given a nonnegative matrix X (N by n) and a reduced rank r, we find two non-negative matrices H (of dimensionality N by r) and R (r by n), visualized as follows

$$\underset{X(N\times n)}{\begin{bmatrix}\cdots & \cdots & \cdots & \cdots\\ \cdots & \cdots & \cdots & \cdots\\ \cdots & \cdots & \cdots & \cdots\\ \cdots & \cdots & \cdots & \cdots\end{bmatrix}} \approx \underset{H(N\times r)}{\begin{bmatrix}\cdots & \cdots\\ \cdots & \cdots\\ \cdots & \cdots\\ \cdots & \cdots\end{bmatrix}} \underset{R(r\times n)}{\begin{bmatrix}\cdots & \cdots & \cdots & \cdots\\ \cdots & \cdots & \cdots & \cdots\end{bmatrix}} \tag{2.11}$$

Through the decomposition process, we factorize high-dimensional data into low-dimensional data and a weight matrix to achieve dimensionality reduction. The objective is to minimize the error between the original data and the data reconstructed by matrix H and R, i.e., we minimize the data loss encountered in the dimensionality reduction process:

$$\begin{aligned} f &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{n}(X_{ij} - (HR)_{ij})^2 \\ H, R &= \arg_{H,R}\min f \end{aligned} \tag{2.12}$$

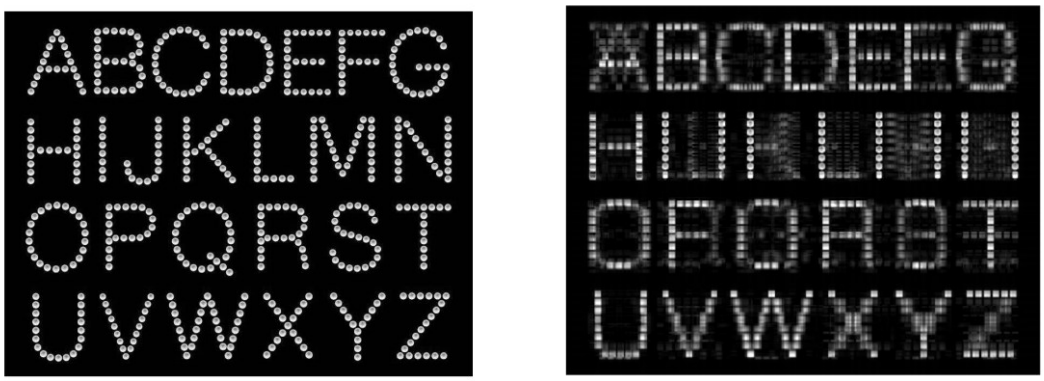
and the iterative process of finding optimal H and R is shown as follows [75], [76].

$$\begin{aligned} H^{iter+1} &= H^{iter}\frac{(X(R^{iter})^T)}{(H^{iter}R^{iter}(R^{iter})^T)} \\ R^{iter+1} &= R^{iter}\frac{((H^{iter})^T X)}{((H^{iter})^T H^{iter}R^{iter})} \end{aligned} \tag{2.13}$$

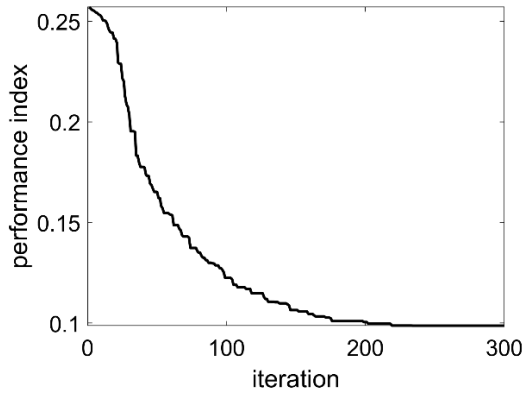
where *iter* stands for the index of iterations.

As an illustrative example, an original image is shown in Fig. 2.2(a) with a size 680×870. By using NMF, it is decomposed into a 680×20 basis matrix H and a 20×870 weight matrix R.

Fig. 2.2(b) shows the reconstruction picture by the two decreasing-dimensionality matrices HR, which shows that NMF can reduce the dimensionality of the data while retaining the information of the original data. Fig. 2.2(c) depicts the performance index changes in successive iterations.



(a) (b)



(c)

Fig. 2.2. An example of values of the performance index (2.9) in successive iterations of the algorithm: (a) Original image, (b) Reconstructed image, (c) Relation between iteration and performance index.

In Fig. 2.3, we show another example with a small image data whose size is 13×60 while the iteration number is 60. Fig. 2.3(c) is the basis matrix and weight matrix.

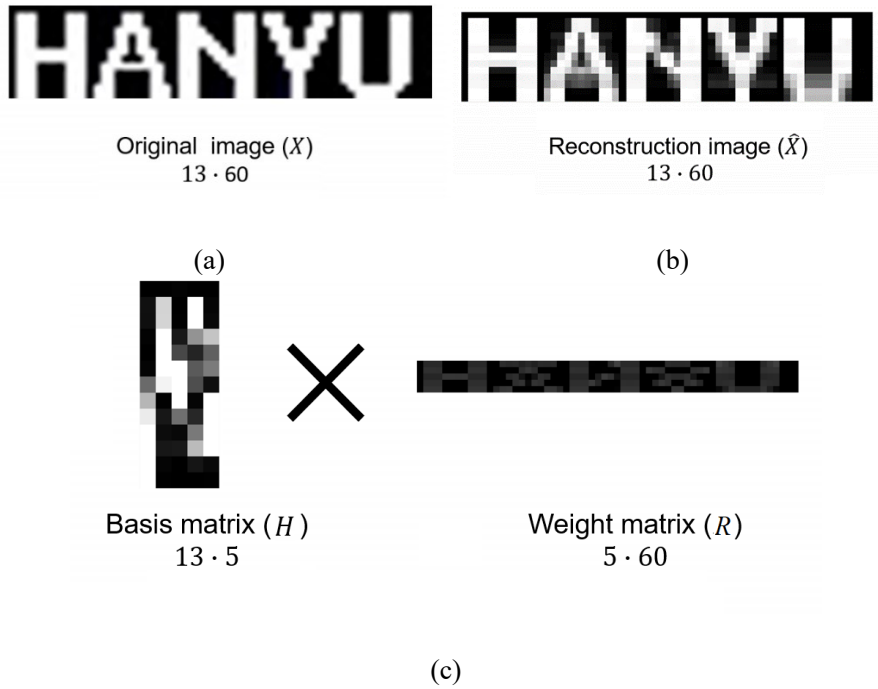


Fig. 2.3. Resulting matrices of NMF. (a) Original data. (b) Reconstructed data. (c) Basis matrix and weight matrix.

The main advantage of this method is that it is simple, faster, and more convenient to process large-scale data. Recently, NMF has been widely used in many fields, e.g., image clustering and labeling [77], [78], face recognition and characterization [79]–[81], social network clustering [82], speech processing [83], [84] and biomedical engineering [85], etc. Especially in recent years, there have been algorithmic developments in matrix factorization. Arefin proposed a new iterative method to replace the least square method in NMF, which can effectively increase the convergence rate [86]. A dual embedding regularized NMF was proposed in [87], where the developed model can effectively improve the performance of matrix factorization. Huang proposed an enhanced incremental NMF model that can effectively reduce the memory requirements without loss of

performance [88]. The multi-objective sparse non-negative matrix factorization proposed in [89] improved the performance of the model.

2.4 Ensemble learning

The underlying idea of ensemble learning is to realize modeling through a series of multiple independent models. This concept was first proposed by Breiman [90]. In contrast to traditional machine learning, the ensemble algorithm focuses on the integration of the results obtained by the independent models, which has been widely investigated and applied [14], [70], [91]–[102]. Ensemble learning involves two main categories: bagging and boosting. Bagging is a way to decrease the variance in the prediction by generating additional data for training from the dataset using combinations with repetitions to produce multi-sets of the original data. Boosting is an iterative technique that adjusts the weight of an observation based on the last classification.

Bagging (bootstrap aggregating) is an ensemble idea that is to create several subsets of data from samples chosen randomly with replacement. Each collection of subset data is used to train the model separately, and then aggregate the sub-models together. One of the most common algorithms is the random forest model proposed by Ho in 1998 [103]. As the name suggests, Random Forest is a forest constructed in a random way, which is composed of many trees. Every single tree is calculated based on a certain number of features randomly selected from the dataset to obtain results. For different trees, they will have their own unique results, and the random forest obtains the result by the majority rule. Therefore, in the random forest algorithm, the greater the number of trees, the better the generalization result.

At present, bagging has been widely used. In the field of fuzzy rules, ensemble learning has been applied to assess a driving style. The fusion of fuzzy rule-based models led to the increase in

accuracy of the evaluation to 94% [91]. Dieu [64] proposed a tree ensemble algorithm based on a fuzzy rule-based model to predict flash floods. Hu designed the bagging and boosting mechanisms for assembling fuzzy rule-based models and demonstrated that the performance of the ensemble model was superior to the traditional single model for most datasets [69]. The distributed and hierarchically driven ways of rule development were discussed in [70]. Some augmentations of the resulting models with the aid of information granules were investigated in [14], [92].

Boosting is an iterative technique where multiple sub-datasets are sampled, and the model is trained on each sub-dataset separately. The idea can be traced back to the studies carried out by Dasarathy and Sheela [104]. The difference between boosting and bagging is that each model of boosting cannot be operated in parallel but is completed sequentially. One of the representatives boosting techniques is the gradient boosting algorithm. As an optimization algorithm, it improves the overall performance of the model by optimizing the loss function, commonly used in regression or classification and other problems. During computation, optimization is based on residuals. In a nutshell, in the gradient boosting algorithm, each iteration is revised based on the results of the previous round. In addition, an adaptive boosting (Adaboost) algorithm is also widely used. It was first proposed in [105], using an adaptive way of focusing attention on incorrectly classified data by associating them with heavier weights such that the designed model (classifier) is made more focused on such data. The approach proceeds in an iterative way, and the way of re-weighting data is repeated until some stopping criterion has been satisfied. This design strategy is applied in conjunction with various models including decision tree [96], SVM [95], naive Bayes [106], K-means [97], etc.

The boosting technique has been continuously studied to improve the accuracy of the model. For example, Pratima proposed to use it to predict solar radiation, where an experimental study

confirmed that, compared with the traditional random forest algorithm, the prediction error rate is reduced by 40%, and the training time of the model is greatly reduced [107]. Zhang proposed a model that uses the gradient boosting model by analyzing images to predict pm2.5 and improve the recognition accuracy [108].

2.5 Gradient-based learning algorithm

The concept of gradient descent could be traced back to 1847, first proposed by Cauchy [109], which aims to minimize the loss function. In other words, it is a process of finding the extreme value of the function. In subsequent developments, the concept of gradients has been frequently refined and applied to multiple domains [110]–[112]. Denote $f(\theta)$ as the objective function f with θ being the parameter and the objective is to minimize this function by changing the value of the parameter. The values of θ are iteratively updated by using the gradient of the objective function $\nabla_\theta f(\theta)$

$$\theta_{iter+1} = \theta_{iter} - \alpha \nabla_\theta f(\theta) \tag{2.14}$$

where α is a positive learning rate.

As gradient descent has been intensively studied, many improved and enhanced algorithms have been proposed, including the Adaptive moment Estimation (ADAM) algorithm [113]. The generic learning scheme is augmented by admitting adaptive learning rates associated with individual optimized parameters [113]. Compared with the basic gradient descent algorithm, ADAM's advantage is that the learning rate of individual parameters is adjusted according to the recorded historical learning rate. For example, in the optimization process, if one variable has been

optimized to the optimal state, but the other variables need further optimization, it is obviously unreliable to use a uniform learning rate for learning. The ADAM algorithm can effectively avoid this problem. In what follows, we briefly elaborate on the details of the algorithm.

By denoting $\boldsymbol{de} = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ coming from gradient descent, we complete the following computing:

$$\begin{gathered}
\boldsymbol{de}(iter) = \nabla_{\boldsymbol{\theta}} f_{iter}(\boldsymbol{\theta}); \\
\boldsymbol{m}(iter+1) = \beta_1\ \boldsymbol{m}(iter) - (1-\beta_1)\ \boldsymbol{de}(iter); \\
\widehat{\boldsymbol{m}}(iter+1) = \boldsymbol{m}\frac{iter}{1-\beta_1{}^{iter}}; \\
\boldsymbol{e}(iter+1) = \beta_2\ \boldsymbol{e}(iter) - (1-\beta_2)\ \boldsymbol{de}^2(iter); \\
\hat{\boldsymbol{e}}(iter+1) = \boldsymbol{e}\ (iter)/(1-\beta_2{}^{iter}); \\
\boldsymbol{\theta}(iter+1) = \boldsymbol{\theta}(iter) - \alpha\widehat{\boldsymbol{m}}(iter+1)/(\sqrt{\hat{\boldsymbol{e}}\ (iter+1)} + \delta).
\end{gathered} \tag{2.15}$$

where δ assumes a small positive value that prevents division by zero, $\boldsymbol{de}^2$ stands for the element-wise square of $\boldsymbol{de}$. $\boldsymbol{m}$ is the first moment estimate and $\boldsymbol{e}$ is the second raw moment estimate. β_1 and β_2 are the exponential decay rates for the first and second moment estimates, respectively, controlling their moving averages. However, those moving averages are initialized as 0's, which causes that the moment estimates are biased towards 0. Considering this issue, the algorithm is improved as follows

$$\begin{gathered}
\boldsymbol{de}(iter) = \nabla_{\boldsymbol{\theta}} f_{iter}(\boldsymbol{\theta}); \\
\boldsymbol{m}(iter+1) = \beta_1\ \boldsymbol{m}(iter) - (1-\beta_1)\ \boldsymbol{de}(iter); \\
\boldsymbol{e}(iter+1) = \beta_2\ \boldsymbol{e}(iter) - (1-\beta_2)\ \boldsymbol{de}^2(iter); \\
\alpha(iter+1) = \alpha(iter)\sqrt{(1-\beta_2{}^{iter})}/(1-\beta_1{}^{iter}); \\
\boldsymbol{\theta}(iter+1) = \boldsymbol{\theta}(iter) - \alpha(iter+1)\ \boldsymbol{m}(iter+1)/(\sqrt{\boldsymbol{e}(iter+1)} + \delta).
\end{gathered} \tag{2.16}$$

2.6 Cross validation in the development of the model

Usually, when building a machine learning model, we need to train the model with known data and test the trained model to verify its performance. Therefore, the data are usually divided into the training set and the testing set. In order to ensure the validity of the experiment, the process of separating the data is often done randomly. But this randomness leads to the generation of sampling error. Therefore, researchers introduce the concept of the generalization error, which is a tool to explain the generalization performance of learning algorithms [114]. The generalization error can be decomposed into bias, variance, and noise as follows

$$generalization\ error = bias^2 + variance + noise \tag{2.17}$$

where $bias$ is an error from erroneous assumptions in the learning algorithm and $variance$ is an error from sensitivity to small fluctuations in the training set.

In order to minimize the generalization error, a cross-validation method is often used in the experiment process. The advantage of cross-validation is that when the number of datasets is limited, the cross-validation method can effectively expand the dataset, thereby verifying the generalization ability of the model to a certain extent. In this study, we use a relatively common method, namely *K*-fold cross validation, which is a dynamic validation method that reduces the impact of data partitioning. The specific steps are as follows: first, divide the dataset into *K* parts; then use 1 of the *K* copies as the testing set, and group all others as the training set. After *K* times, every part has been used as a testing set. Note that we calculate the average values of *K* times as the model output [115], [116].

2.7 Performance indexes

There are a plethora of methods for evaluating a model. Here we focus on the following approaches that have been selected as evaluation criteria in this study.

Mean-square error (MSE): This is a common method to calculate the error Q between the estimated value (*est*) produced by the model and the actual value (*act*), which measures how close the fitted line is to the actual data points. The smaller the mean squared error, the closer the fit of the model is to the dataset:

$$Q = \frac{1}{N}\sum_{k=1}^{N}(est_k - act_k)^2 \tag{2.18}$$

where N denotes the number of data.

Root mean square error (RMSE): It measures the average magnitude of the error and directly interpreter it in terms of measurement units, as shown below:

$$RMSE = \sqrt{MSE} \tag{2.19}$$

Mean absolute percentage error (MAPE): This is a measure of prediction accuracy [117] and is commonly used as a loss function for a regression problem, which is concerned with the ratio of error between *est* and *act* to *est*. Its advantage is that the presented results are ratios rather than absolute numbers, making it easy to compare models across units of magnitude. The details are shown as follows:

$$Q = \frac{1}{N}\left(\sum_{k=1}^{N}\left|\frac{est_k - act_k}{est_k}\right|\right) \tag{2.20}$$

For classification problems, cross-entropy is also often used as an effective evaluation tool. Cross entropy represents two probability distributions $\boldsymbol{p}$ and $\boldsymbol{q}$. $\boldsymbol{p}$ is the represent concern of the data and $\boldsymbol{q}$ is generated by the model. The representation is described as follows

$$Q = \sum_{i} \boldsymbol{p}_i \log\left(\frac{1}{\boldsymbol{q}_i}\right) \tag{2.21}$$

In this study, we focus on regression problems and absolute error values and thus choose RMSE as the main evaluation criterion.

2.8 Information granules

Information granules were proposed by Zadeh in 1979 [118]. They are collections of entities (elements), usually originating at the numeric level, which is arranged together due to their similarity, functional adjacency and indistinguishability or alike. Given the similarity function to quantify the closeness between the samples, these data are clustered into certain granules, categories or classes [119]. Information granules are constructed in various formalisms: intervals, fuzzy sets, rough sets, etc. The process of forming information granules is referred to as information granulation [120].

When we design information granules on the basis of data, we need an effective design method. Thus, the principle of justifiable granularity was proposed as a carrier of granular computing [121]–[124], in which a justifiable information granule (interval) is expected to find to

contain the data as much as possible but without being too general. Pedrycz and Homenda abstracted a complex problem and obtained the interval through two parameters: coverage and specificity [125]. As coverage focuses on the extent of data being included and specificity is the extent that how 'specific' the interval is, which results in these two performance indexes being of a conflicting nature. The increase in coverage implies a decrease in specificity and vice versa. The optimal interval is determined by maximizing the product V of coverage (cov) and specificity (sp).

$$V = cov \cdot sp \tag{2.22}$$

We briefly illustrate it with an example. For one-dimensional data $(x_1, x_2, \dots, x_N)$, we can build an information granule interval taking the form $A = [a, b]$. The coverage and specificity are shown as follows:

Coverage This measure is a simple count of data included in the corresponding intervals of respective elements.

$$cov(A) = \frac{1}{N} card\{x_k | x_k \in A\} \tag{2.23}$$

where $card(.)$ denotes the cardinality of A.

Specificity This index quantifies the precision of the result. Qualitatively it can be regarded as a decreasing function of the length of the interval, see (2.24). Such a function has to satisfy some obvious boundary conditions: an interval reduced to a single numeric value comes with the highest specificity of one and the interval expanded to the entire unit interval has the zero-

specificity value, in other words, specificity indicates the degree of specificity of the data covered by the interval.

$$sp(A) = 1 - \frac{|b - a|}{range} \tag{2.24}$$

where $range = \max(x_1, x_2, \dots, x_N) - \min(x_1, x_2, \dots, x_N)$.

2.9 Conclusions

This chapter covers the main methods and algorithms that are critical in the experimental design. First, we introduce the method, fuzzy clustering and fuzzy rule-based model used throughout the paper. Then we expose the idea of non-negative matrix factorization that will be applied in Chapters 3 and 4. Next, we focus on ensemble learning that will be used in Chapter 5. Finally, we briefly summarize the data distribution methods, model evaluation methods, and the concept of information granules used in the experiments.

Chapter 3 Fuzzy Relational Matrix Factorization and Its Granular Characterization in Data Description

It is evident that decreasing the dimensionality of data is an effective way to enhance the effectiveness of the design of models and increase their performance. Dimensionality reduction has been a central area of studies in system modeling and there have been a number of well-established methods such as autoencoders [126]–[130], nonnegative matrix factorization [131]–[134], Principal Component Analysis (PCA) [135]–[137]. In this chapter, we will introduce the single-level and two-level factorization to reduce data dimensionality which has better performance and stronger interpretability than traditional non-negative matrix factorization.

3.1 Single-level fuzzy relation factorization

A collection of data organized in an N by n matrix $X = [x_{ij}], i = 1, 2, \dots, N, j = 1, 2, \dots, n$, X is composed of vectors located in the $[0,1]^n$ hypercube. As we said in Chapter 2, the idea was to decompose it into two matrices. Here, we are making a substantial departure by treating these matrices as fuzzy relations and applying the calculation of fuzzy relations such as *s-t* decompositions. Therefore, the factorization problem is expressed in the formal way as follows:

$$\hat{X} = rel(H, R) \tag{3.1}$$

where $H = [h_{ij}]$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, p$, with p being the number of decreased dimensionality and $p \ll n$, is an N by p matrix of reduced data and $R = [r_{ij}], i = 1, 2, \dots, p, j =$

$1, 2, \ldots, n$, is a p by n matrix representing the fuzzy relation of dimensionality, and *rel* stands for some relational operators.

The dimensionality of H is lower than that of the original data coming as a result of the reduction process. The objective of the factorization problem is to determine H and R such that $\hat{X}$ is made as close to X as possible. Realistically, the equality $\hat{X} = X$ is not feasible and the optimization is about minimizing some distance $||.||$ between X and $\hat{X}$, say $\left\|X - \hat{X}\right\|^2$. This is realized by determining suitable fuzzy relations R and H. Proceed with the detailed processing behind the relational factorization. The relational operator *rel* can assume two main forms where the underlying computing engages triangular norms (t) and triangular t-conorms (s). The following two composition operators coming from the relational calculus are encountered:

s-t composition

$$\hat{X} = H \circ R \tag{3.2}$$

t-s composition

$$\hat{X} = H \cdot R \tag{3.3}$$

where '$\circ$' and '$\cdot$' are the corresponding *s-t* and *t-s* composition operators, respectively.

Let us introduce the following vector notations:

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix};\ H = \begin{bmatrix} h_1^T \\ h_2^T \\ \vdots \\ h_N^T \end{bmatrix} \tag{3.4}$$

where $\boldsymbol{x}_i$ is a vector located in the n-dimensional unit hypercube, say $x_i = [x_{i1}, x_{i2}, \ldots, x_{in}]$ and $\boldsymbol{h}_i$ is a vector positioned in the p-dimensional unit hypercube, namely $h_i = [h_{i1}, h_{i2}, \ldots, h_{ip}]$, $i = 1, 2, \ldots, N$.

Furthermore, let us arrange R in the following way:

$$R = [r_1, r_2, \ldots, r_n];\ dim(r_j) = p;\ r_j = \begin{bmatrix} r_{1j} \\ r_{2j} \\ \vdots \\ r_{pj} \end{bmatrix} \tag{3.5}$$

Accordingly, the factorization problem is transformed into a system of relational equations, where the $(i,j)^{th}$ element of $\hat{X}$, afternote by x_{ij}, is expressed in the form for (3.2):

$$\hat{x}_{ij} = \boldsymbol{h}_i \circ \boldsymbol{r}_j \tag{3.6}$$

where $\circ$ is *s-t* factorization.

In terms of the individual elements of R and H, the above expression reads as follows:

$$\hat{x}_{ij} = \underset{l=1}{\overset{p}{S}} \left(h_{il}\ t\ r_{lj}\right) \tag{3.7}$$

For (3.3)

$$x_{ij} = h_i \cdot r_j;\ \hat{x}_{ij} = \underset{l=1}{\overset{p}{T}}\left(h_{il}\ s\ r_{lj}\right) \tag{3.8}$$

where $i = 1, 2, \ldots, N, j = 1, 2, \ldots, n$, and $l = 1, 2, \ldots, p$.

In light of triangular norms and conorms, the processing is evidently logic-oriented because the operator between the matrices is a relation.

To solve the above factorization problem, one has determined simultaneously the fuzzy relation *R* and the data relation H. There are two phases of the design process, namely the structural optimization followed by the parametric optimization. The former involves the optimization of dimensionality (p) of the fuzzy relation and the choice of *t*-norms and *t*-conorms. The lather is concerned with the adjustment of the entries of the fuzzy relation R. The performance index Q used in the optimization process is a sum of squared errors between the corresponding X and $\hat{X}$:

$$Q = \sum_{i=1}^{N}\sum_{j=1}^{n}(x_{ij} - \hat{x}_{ij})^2, i = 1, 2, \ldots, N, j = 1, 2, \ldots, n. \tag{3.9}$$

Consider formulas (3.7) and (3.8), for the factorization problem (3.9), one develops the following gradient-based iterative scheme (*iter* stands for the index of the iterative process):

$$h_{bc}(iter + 1) = h_{bc}(iter) - \alpha\nabla_{h_{bc}}Q(iter) \tag{3.10}$$

$b = 1, 2, \ldots, N;\ c = 1, 2, \ldots, p$.

$$r_{cd}(iter + 1) = r_{cd}(iter) - \alpha\nabla_{r_{cd}}Q(iter) \tag{3.11}$$

$c = 1, 2, \dots, p;\ d = 1, 2, \dots, n.$

where α is a positive learning rate. During the iterative process, the initial value is randomly generated, and we limit the value generated by the iterative process is limited to the interval [0, 1]. Proceed with details, and the pertinent formulas are as follows:

$$\nabla_{h_{bc}} Q = 2 \sum_{j=1}^{n} (x_{bj} - \hat{x}_{bj}) \frac{\partial \hat{x}_{bj}}{\partial h_{bc}};$$
$$\nabla_{r_{cd}} Q = 2 \sum_{i=1}^{N} (x_{id} - \hat{x}_{id}) \frac{\partial \hat{x}_{id}}{\partial r_{cd}} \tag{3.12}$$

The detailed formulas for the calculations of the gradient (3.12) are completed below.

s-t composition:

$$\nabla_{h_{bc}} Q = 2 \sum_{j=1}^{n} (x_{bj} - \hat{x}_{bj}) \frac{\partial \left\{ \underset{l=1}{\overset{p}{S}} (h_{bl}\ t\ r_{lj}) \right\}}{\partial h_{bc}}$$
$$= 2 \sum_{j=1}^{n} (x_{bj} - \hat{x}_{bj}) \frac{\partial \left\{ \left[\underset{l=1, l \neq c}{\overset{p}{S}} (h_{bl}\ t\ r_{lj}) \right]\ s\ (h_{bc}\ t\ r_{cj}) \right\}}{\partial h_{bc}}$$

(3.13)

$$\nabla_{r_{cd}} Q = 2 \sum_{i=1}^{N} (x_{id} - \hat{x}_{id}) \frac{\partial \left\{ \underset{l=1}{\overset{p}{S}} (h_{il}\ t\ r_{ld}) \right\}}{\partial r_{cd}}$$
$$= 2 \sum_{i=1}^{N} (x_{id} - \hat{x}_{id}) \frac{\partial \left\{ \left[\underset{l=1, l \neq c}{\overset{p}{S}} (h_{il}\ t\ r_{ld}) \right]\ s\ (h_{ic}\ t\ r_{cd}) \right\}}{\partial r_{cd}}$$

t-s composition:

$$\nabla_{h_{bc}} Q = 2\sum_{j=1}^{n}(x_{bj} - \hat{x}_{bj})\frac{\partial\left\{\overset{p}{\underset{l=1}{T}}(h_{bl}\ s\ r_{lj})\right\}}{\partial h_{bc}}$$

$$= 2\sum_{j=1}^{n}(x_{bj} - \hat{x}_{bj})\frac{\partial\left\{\left[\overset{p}{\underset{l=1,l\neq c}{T}}(h_{bl}\ s\ r_{lj})\right]\ t\ (h_{bc}\ s\ r_{cj})\right\}}{\partial h_{bc}};$$

(3.14)

$$\nabla_{r_{cd}} Q = 2\sum_{i=1}^{N}(x_{id} - \hat{x}_{id})\frac{\partial\left\{\overset{p}{\underset{l=1}{T}}(h_{il}\ s\ r_{ld})\right\}}{\partial r_{bc}}$$

$$= 2\sum_{i=1}^{N}(x_{id} - \hat{x}_{id})\frac{\partial\left\{\left[\overset{p}{\underset{l=1,l\neq c}{T}}(h_{il}\ s\ r_{ld})\right]\ t\ (h_{ic}\ s\ r_{cd})\right\}}{\partial r_{cd}}.$$

If we confine ourselves to some particular *t*-norm and conorm, say the product and probabilistic sum, the formulas are written down in the following way:

s-t composition:

$$\nabla_{h_{bc}} Q = 2\sum_{j=1}^{n}\left(x_{bj} - \hat{x}_{bj}\right) r_{cj}\left(1 - \left[\overset{p}{\underset{l=1,l\neq c}{S}}\left(h_{bl} r_{lj}\right)\right]\right);$$

$$\nabla_{r_{cd}} Q = 2\sum_{i=1}^{N}(x_{id} - \hat{x}_{id})\, y_{ic}\left(1 - \left[\overset{p}{\underset{l=1,l\neq c}{S}}(h_{il} r_{ld})\right]\right) \quad (3.15)$$

t-s composition:

$$\nabla_{h_{bc}} Q = 2\sum_{j=1}^{n}(x_{bj} - \hat{x}_{bj})\left[\underset{l=1,l\neq c}{\overset{p}{T}}(h_{bl} + r_{lj} - h_{bl}r_{lj})\right](1 - r_{cj});$$

$$\nabla_{r_{cd}} Q = 2\sum_{i=1}^{N}(x_{id} - \hat{x}_{id})\left[\underset{l=1,l\neq c}{\overset{p}{T}}(h_{il} + r_{ld} - h_{il}r_{ld})\right](1 - h_{ic}) \tag{3.16}$$

For other triangular norms and conorms, the general learning scheme exhibits the same structure; however, the computing details are modified to accommodate the derivatives of the triangular norms and conorms of interest.

In both factorization schemes involving R and H, for practical purposes, the generic learning scheme is augmented by admitting adaptive learning rates associated with individual optimized parameters as discussed in the ADAM stochastic optimization [113]. In what follows, we briefly elaborate on the details of the algorithm.

Denote by $\boldsymbol{\theta}$ a vector of parameters composed of elements of H and R organized in a linear way in the form of a single vector. The resulting gradient vector $\boldsymbol{de} = \begin{bmatrix} \nabla_H Q \\ \cdots\cdots \\ \nabla_R Q \end{bmatrix}$ is $(NP + pn)$ dimensional. We complete the following computation with ADAM (2.16).

Logic interpretation

As noted, the composition operators come with a sound logic-oriented interpretation. Consider the i^{th} row of matrix H and the j^{th} column of the matrix R. The original input x_{ij} is factorized as described by (3.7) and (3.8). Then recall that any *t*-norm realizes an *and* operator while *t*-conorm realizes an *or* operator. We have:

$$x_{ij} = \left[\left(h_{i1}\ and\ r_{1j}\right)\right]\ or\ \ldots\ or\ \left[\left(h_{ip}\ and\ r_{pj}\right)\right] \tag{3.17}$$

Alternatively, we emphasize the contribution of entries of $\boldsymbol{h}_i$ to the result x_{ij} by rewriting (3.17) as:

$$x_{ij} = h_{i1}|_{r_{1j}} or\ h_{i2}|_{r_{2j}} or \ldots or\ \ h_{ip}|_{r_{pj}} \tag{3.18}$$

In light of the boundary properties of *t*-norms, higher values of $r_{1j}, r_{2j}, \ldots, r_{pj}$ highlight the more essential contributions of the corresponding components of $\boldsymbol{h}_i$ to the formation of x_{ij}.

The realization of x_{ij} completed through the *t-s* composition is achieved by looking at those terms in the expression:

$$x_{ij} = h_{i1}|_{r_{1j}} and\ h_{i2}|_{r_{2j}} and \ldots and\ \ h_{ip}|_{r_{pj}} \tag{3.19}$$

where the entries are associated with lower values of $r_{1j}, r_{2j}, \ldots, r_{pj}$. It means that in this logic expression, the essential components are the corresponding h with lower values of r.

Consider the fuzzy relation R, which quantifies the associations between variables present in the original and reduced data spaces. For the *s-t* composition, the values of R closer to 1 exhibit high relevance. Given R, where one has entries of a certain column of R close to 1, these high values of R identify the variables in the reduced space that associated with a certain variable in the input space. In the case of the *t-s* composition, the entries of the fuzzy relation close to zero are of interest to identify the essential logic dependencies between H and X. Formally, a single level factorization algorithm is structured as follows:

Algorithm1: Single level fuzzy relation factorization

Input: Data matrix $X \in R^{N \times n}$, parameters $\alpha, \beta_1, \beta_2, \delta$, maxiteration
Output: Data matrix $\hat{X}$ and R
Initialize H and R as arbitrary positive $N \times p, p \times n$matrices, respectively
for $p \leftarrow$1 to n **do**
    while iter<maxiteration **do**
        Update H and R with ADAM algorithm (2.13)
        Construct $\hat{X} = H \cdot R$ or $\hat{X} = H \circ R$
    end
end
Return the output $\hat{X}$ and R

3.2 Two-level fuzzy relation factorization

A two-level factorization is the extension of the above constructs. We further factorize H. In the consecutive stages, the resulting logic expression results are shown as follows. The architectures of relational factorization discussed so far exhibit a single-level structure being either realized by the *s-t* or *t-s* composition. Their underlying logic structure is either guided by the *and* or *or* logic aggregation. The factorization can be realized through a two-level architecture (where the logic expressions are put together). Two variants of the structure are considered:

$$\hat{X} = H \cdot R;\ \widehat{H} = Z \circ G \tag{3.20}$$

and

$$\hat{X} = H \circ R; \widehat{H} = Z \cdot G \tag{3.21}$$

In this two-level structure, one first factorizes X producing the data of reduced dimensionality H and the fuzzy relation R. Then, H is again factorized using the *s-t* composition, resulting in further reduction of H thus yielding Z and producing the fuzzy relation G.

In terms of the underlying logic exercised here, one has a factorization completed by the *and* aggregation followed by the *or* aggregation or vice versa.

The arrangement of the composition operators following (3.20) and (3.21) is motivated by the logical nature of the processing implied by triangular norms, as shown in Fig. 3.1.

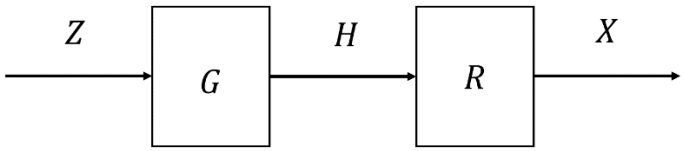


Fig. 3.1. A two-level relational factorization.

In light of the underlying logic processing completed by the *and* and *or* generalized logic expressions, it becomes evident that the values of R and G of Boolean nature are easily interpretable. In contrast, the entries of the fuzzy relation that are positioned close to 1/2 are more difficult to comprehend.

Having this in mind, the learning process can be modified by accommodating a regularization mechanism. Using it, we tend to move the optimized parameters of R and G to be positioned closer 1 or 0. This is accomplished by augmenting the minimized objective function:

$$Q = \sum_{i=1}^{N}\sum_{j=1}^{n}(x_{ij} - \hat{x}_{ij})^2 + \lambda \sum_{i=1}^{p}\sum_{j=1}^{n} \Phi(r_{ij}) \tag{3.22}$$

where $\lambda \geq 0$ and the second term-involving Φ serves as the regularization term. In virtue of the Boolean requirements facilitating interpretability, the function Φ used in the regularization part satisfies $\Phi(0) = \Phi(1) = 0$. We also require that Φ should be an increasing function over $[0, \frac{1}{2}]$, decreasing function over $[\frac{1}{2}, 1]$ and $\Phi(\frac{1}{2}) = 1$ is the maximal value over the unit interval. An example of Φ comes as $\Phi(\xi) = 4\xi(1-\xi)$, where $\xi \in [0,1]$. The overall structure of the two-level decomposition algorithm is outlined below:

Algorithm2: Two level fuzzy relation factorization

Input: Data matrix $X \in R^{N \times n}$, parameters $\alpha, \beta_1, \beta_2, \delta$, maxiteration
Output: Data matrix $\hat{X}$ and R and G
Initialize H, R, Z and G as arbitrary positive $N \times p, p \times n, N \times m, m \times p$ matrices, respectively
for $p \leftarrow$1 to n **do**
    while iter<maxiteration **do**
        Update H and R with ADAM algorithm (2.13)
    end
end
for $m \leftarrow$1 to p **do**
    while iter<maxiteration **do**
        Update Z and G with ADAM algorithm (2.13)
        Construct $\hat{H} = Z \circ G$ or $\hat{H} = Z \cdot G$
        Construct $\hat{X} = H \cdot R$ or $\hat{X} = H \circ R$
    end
end
Return the output $\hat{X}$ and R and G

3.3 Granular relation factorization

Evidently, the factorization process does not lead to the ideal results, which means that the optimized $\hat{X}$ is not equal to X. To quantify this effect, we make the results be expressed as information granules rather than single numeric membership grades. In particular, we consider entries of R to be information granules, in the form of the interval-valued membership grades. Briefly, by admitting interval-valued entries, we arrive at the granular fuzzy relation, $R^{\sim} =$

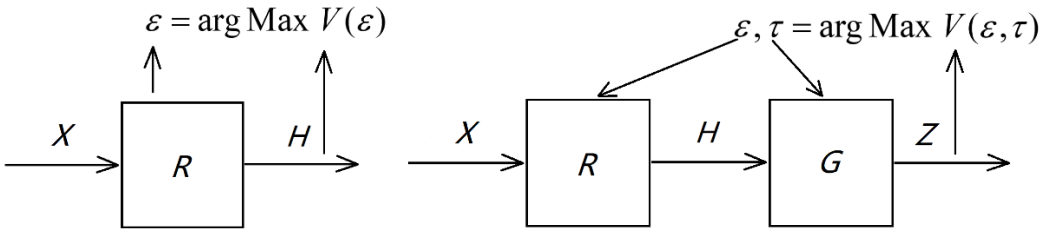
$[R^-, R^+]$. To emphasize the impact of the level of information granularity on the quality of the factorization results, we also use the alternative notation R_ε where ε is referred to as a level of information granularity which are the range from 0 to 1. This in turn makes the factorization result coming in the interval format, say $X^\sim = [X^-, X^+]$. In virtue of monotonicity of *t*-norms and *t*-conorms, we have:

$$X^- = H \cdot R^-, X^+ = H \cdot R^+ \tag{3.23}$$

or

$$X^- = H \circ R^-, X^+ = H \circ R^+ \tag{3.24}$$

The essence of injection of information granularity into the single-level and two-level factorization schemes is illustrated in Fig. 3.2.



(a) (b)

Fig. 3.2. Formation of granular fuzzy relation with the aid of the optimized level of information granularity. (a) a single level factorization structure, and (b) two-level factorization structure.

Through the introduction of the level of information granularity, once the matrices R and G have been determined, their entries are generalized to intervals. In light of the monotonicity of triangular norms, for each element of R_ε, one forms the following bounds $r_{ij}{}^- = r_{ij}(1-\varepsilon)$,

$r_{ij}{}^{+} = min(\, r_{ij}(1+\varepsilon),1)$. Similar to the matrix G_τ, we have $g_{ij}{}^{+} = g_{ij}(1-\tau)$, $g_{ij}{}^{+} = min(\, g_{ij}(1+\tau),1)$, where $\varepsilon \in [0,1]$, $\tau \in [0,1]$.

Consider the two-level structures (3.20) and (3.21). Their granular generalizations are formed by introducing levels of information granularity to the fuzzy relations R and G. For $X^{\sim}$one has:

$$X^{\sim} = (Z \circ G_\tau) \cdot R_\varepsilon \tag{3.25}$$

or

$$X^{\sim} = (Z \cdot G_\tau) \circ R_\varepsilon \tag{3.26}$$

The quality of the granular model is evaluated by taking into account the performance of the information granule $X^{\sim}$ vis-à-vis the original data X. The two criteria, coverage and specificity, are relevant here. Let us briefly recall their interpretation, see Section 2.8. One is interested in assessing how well the data X are "covered" viz. included in $X^{\sim}$. Simultaneously one wishes to quantify how precise (specific) the generated information granule $X^{\sim}$ becomes. These two features are quantified by coverage and specificity as follows.

$$cov = \frac{1}{Nn}\sum_{i=1}^{N}\sum_{j=1}^{n} ind\big(x_{ij}, \big[x_{ij}^{-}, x_{ij}^{+}\big]\big) \tag{3.27}$$

where $ind(x_{ij}, [x_{ij}^{-}, x_{ij}^{+}]) = \begin{cases} 1, \text{ if } x_{ij} \in [x_{ij}^{-}, x_{ij}^{+}] \\ 0, \text{ otherwise} \end{cases}$.

$$sp = \frac{1}{Nn}\sum_{i=1}^{N}\sum_{j=1}^{n}\left(1 - \left|{x_{ij}}^{+} - {x_{ij}}^{-}\right|\right) \tag{3.28}$$

The coverage and specificity are in conflict. The optimization of information granularity is completed by maximizing the performance index $V(\varepsilon) = cov(\varepsilon)\ sp(\varepsilon)$, viz. $\varepsilon_{\text{opt}} = arg\max_{\varepsilon}[cov(\varepsilon)sp(\varepsilon)]$ for single level factorization schemes and $V(\varepsilon, \tau) = cov(\varepsilon, \tau)sp(\varepsilon, \tau)$ which is carried out for the two-layer architecture.

3.4 Experimental studies

The experimental study is concerned with the investigations focusing on the performance of factorization. In all experiments, *t*-norm has been specified as the product whereas the corresponding *t*-conorm is taken as the probabilistic sum. The reported performance index is the average of Q with $\bar{Q} = Q/(Nn)$. In the experiments, we use the ADAM algorithm to carry out optimization. The algorithm runs with the values of the parameters set up as follows: $\alpha = 0.1$, β_1 and β_2 are 0.9 and 0.99, respectively, δ is $1e-8$. Here we set the maximal number of iterations to 300 which is enough to make the result changes flatten between iterations.

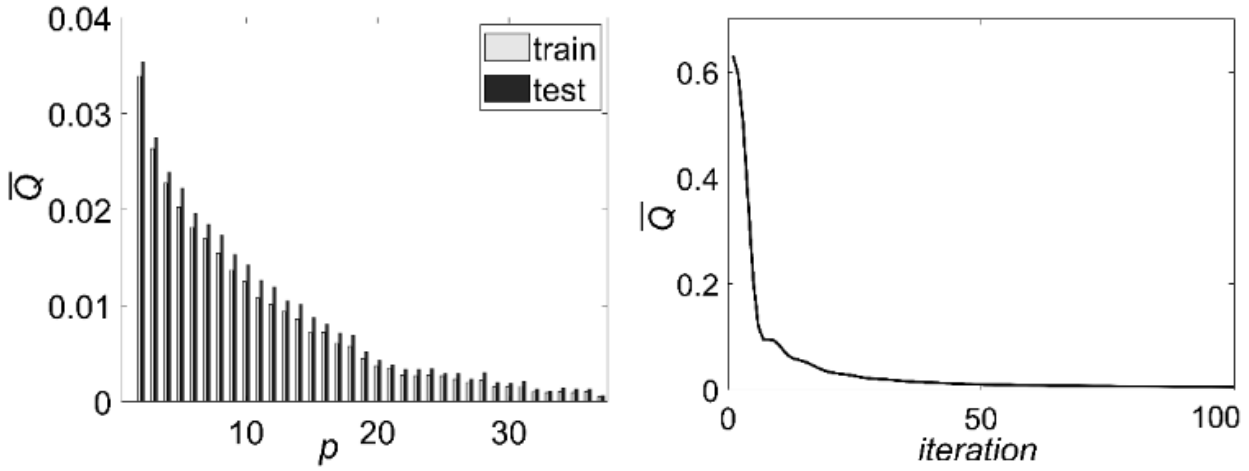
For each dataset, the experiments are repeated 30 times and each time the dataset is randomly split into training and testing subsets: 70% of data are used for training while the remaining 30% of data form the testing set. When carrying out testing, one considers the already determined fuzzy relation R and the original testing data X_{test} to determine the corresponding H_{test} such that we can obtain $\hat{X}_{\text{test}}$ and finally computes the distance between X_{test} and $\hat{X}_{\text{test}}$ to quantify the performance achieved on the testing data.

Experiment 1:

We consider the Boston housing data (https://archive.ics.uci.edu/ml/datasets.php). The data has 12 continuous input variables and a single discrete variable. To build the logic manifestation of these data, for each continuous variable, we use three triangular membership functions with an $1/2$ overlap between the two adjacent fuzzy sets. For the discrete variable, we make it evenly distributed, and for Boolean variable, the one hot encoding is used. There is a variable including only (0, 1), and we maintain this variable as same as the original data. In total, the transformed data are positioned in the 37-dimensional unit hypercube.

The single-level architectures:

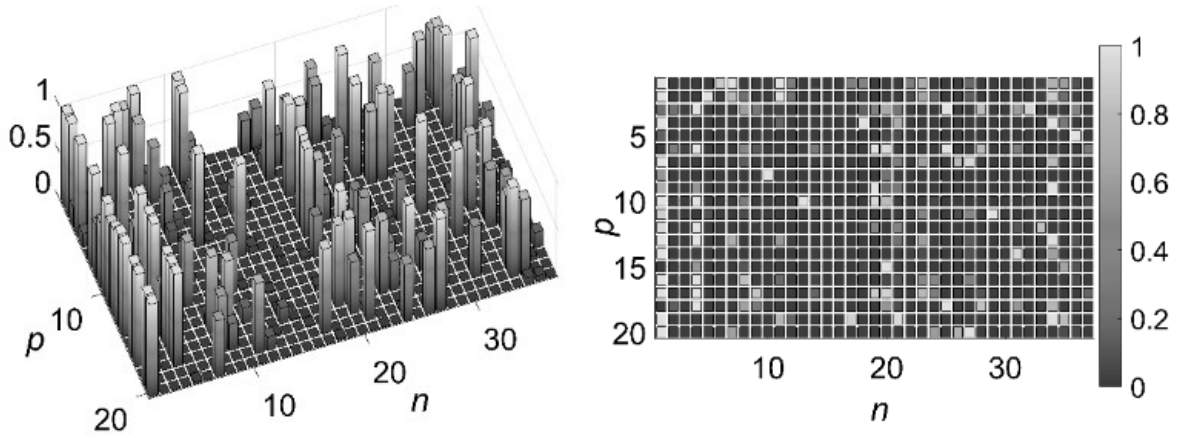
(i) *s-t* factorization. The values of the resulting performance index $\bar{Q}$ obtained from consecutive values of p are displayed in Fig. 3.3(a). The performance improves with the increasing values of p. Apparently $\bar{Q}$ is a monotonically decreasing function of this parameter; however, the decrease becomes slower for higher value of p. The inspection of this curve points at the choice of $p = 20$ as a sound reduced dimensionality. In this case, the convergence of the method is displayed in Fig. 3.3(b). Most of the learning occurs at the beginning of the learning process and this manifests through a significant drop of the values of $\bar{Q}$ that are observed in the first 50 iterations of the learning process.



(a) (b)

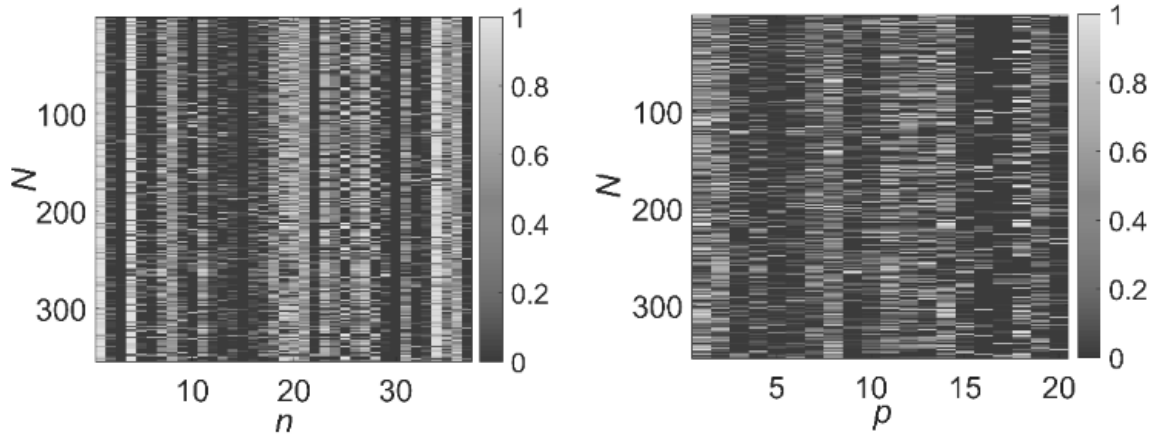
Fig. 3.3. Performance index $\bar{Q}$ for *s*-*t* factorization. (a) $\bar{Q}$ as a function of p; $\bar{Q} = 0.0037$ and 0.0043 for the training and testing data for $p = 20$. (b) convergence of the optimization process for $p = 20$.

Fig. 3.4 displays the values of the optimized fuzzy relation R; the most essential information is the high entries of the relation that assume values close to 1. The figure delivers an interesting view of the relationship revealed by the fuzzy relation. For each column of R there are only a fraction of high entries of the relation; some of the columns come with only low values positioned close to zero. By looking at the rows of R, one can notice that there are a fraction of inputs forming the reduced space. Another interesting view at the reduction process is offered by Fig. 3.5. The matrix of reduced data H is more homogenous (less dispersion of the values of its entries) than the entries of X.



(a) (b)

Fig. 3.4. Fuzzy relation *R* produced through relational factorization. (a) 3-D display; (b) 2-D display.

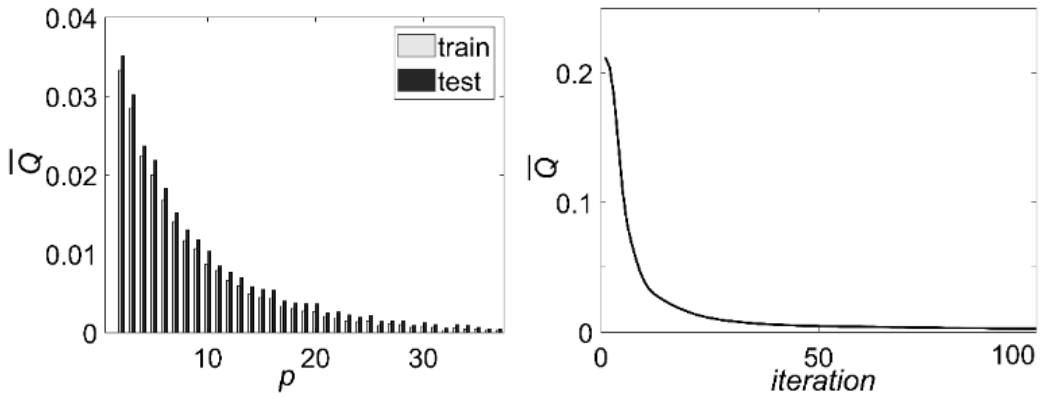


(a) (b)

Fig. 3.5. Display of data. (a) original data *X*; (b) reduced data *H*.

(ii) *t*-*s* factorization.

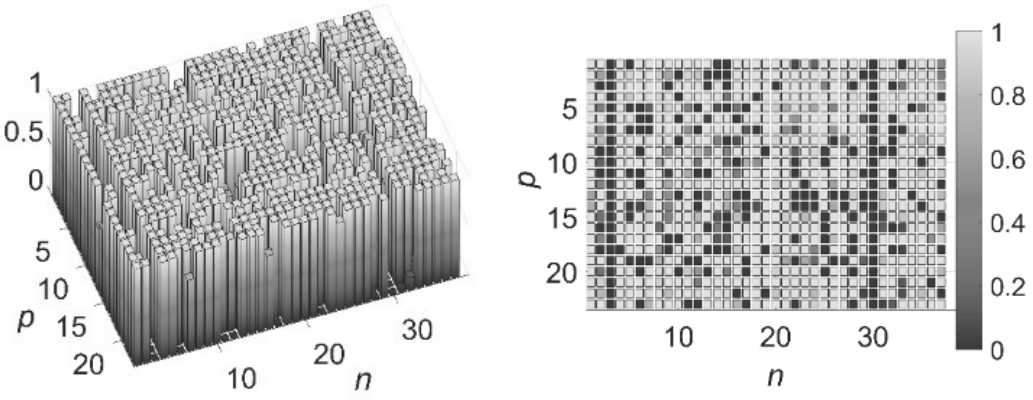
The results of this factorization are collected in Fig. 3.6. As before $\bar{Q}$, is a decreasing function of p with some saturation region, which suggests the choice of $p = 23$ as a cut off value.



(a) (b)

Fig. 3.6. Performance index $\bar{Q}$ obtained for *t-s* model. (a) Performance index $\bar{Q}$ versus varying values of p. $\bar{Q} = 0.0015$ and 0.0022 for the training and testing data for $p = 23$; (b) Convergence of the optimization process for $p = 23$.

Fig. 3.7 contains a bar plot of the values of the optimized fuzzy relation R. The most essential entries are the low entries of the fuzzy relation (assuming values close to 0). By scanning the values of this relation, the number of essential entries is a small fraction of all entries. Some input variables (columns with light color entries) have a very limited contribution to the reduced data. Fig. 3.8 shows the relation H of reduced data. Compared with original data X in Fig. 3.5(a), it shows that some certain variable's limited contribution



(a) (b)

Fig. 3.7. Fuzzy relation R obtained through the *t-s* factorization. (a) 3-D display; (b) 2-D display.

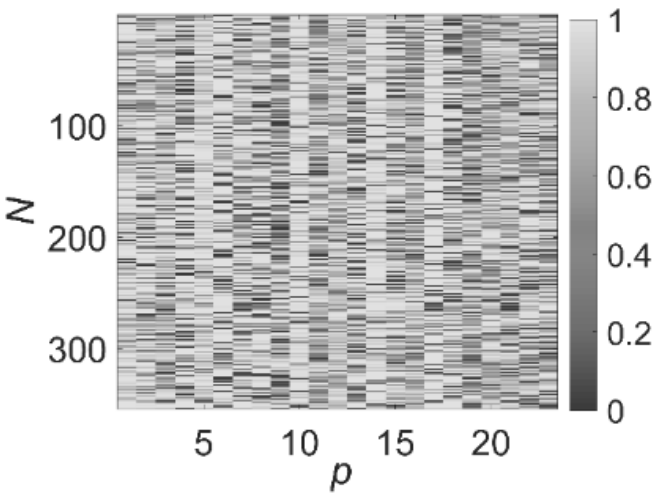


Fig. 3.8. Fuzzy relation H.

For a complete analysis, we considered the standard method of non-negative matrix factorization (NMF). With the same performance index being used, the results are displayed in Fig. 3.9. For comparison with the *s*-*t* composition, considering $p = 20$, one has $\bar{Q} = 0.0056$ and 0.0067 for the training and testing data, respectively. Comparing the values of the performance index on the testing data for these two models, we conclude that the *s*-*t* factorization is better than the NMF producing 35.8% decrease of $\bar{Q}$. For the *t*-*s* factorization, when $p = 23$, the improvement is 46.3% in comparison with the results produced by the NMF method.

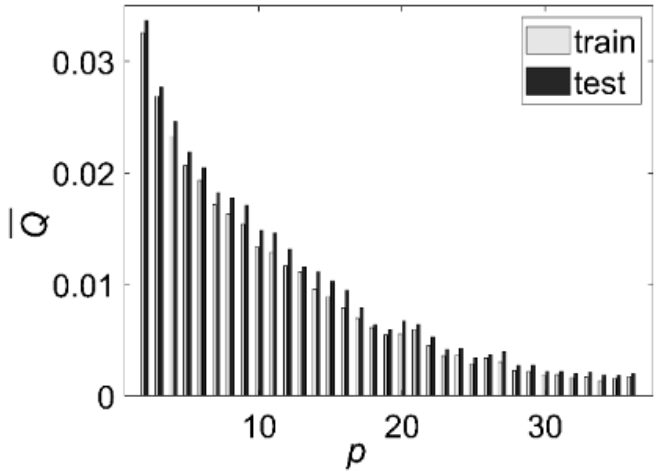
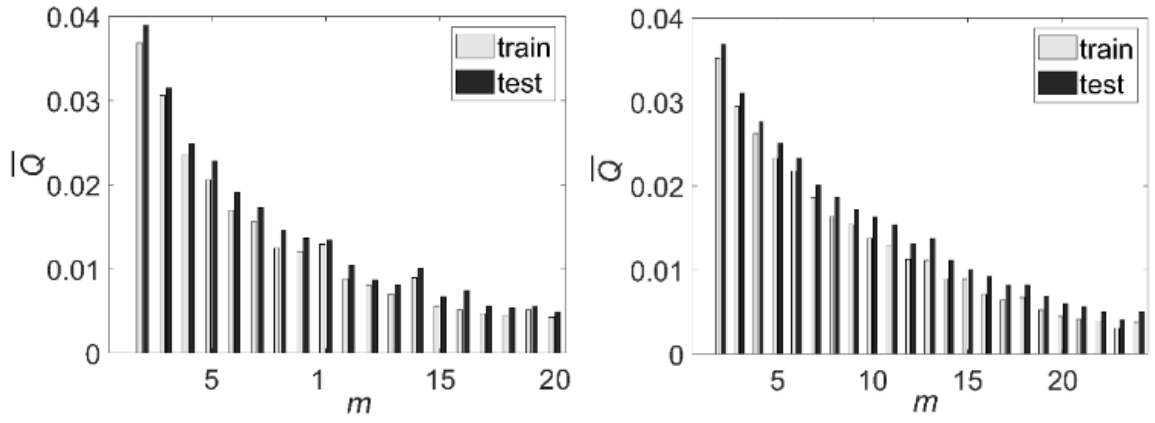


Fig. 3.9. $\bar{Q}$ as a function of p obtained for the NMF algorithm.

Two level relational factorization:

In the construction of the two-level factorization, we follow the method described in Section 5. As in the previous experiment, the optimal values of p associated with the *s-t* and *t-s* factorization are equal to 20 and 23, respectively. Then H is used as the data for further factorization and the optimization is carried out for successive values of m. The obtained results are presented in Figs. 3.10(a) and 3.10(b). The optimal values of m are 14 and 16 for the *s-t* and *t-s* factorization, respectively. In Fig. 3.11, we show the relation G obtained for the optimal value of m. For Figs. 3.11(a) and (b), the most essential are the low entries of the fuzzy relation. In Figs. 3.11(c) and (d), each column of G has only a fraction of high entries of the relation. Those input variables with dark entries exhibit the limited contribution to the reduced data.



(a) (b)

Fig. 3.10. $\bar{Q}$ as a function of m. (a) *s-t* factorization. (b) *t-s* factorization.

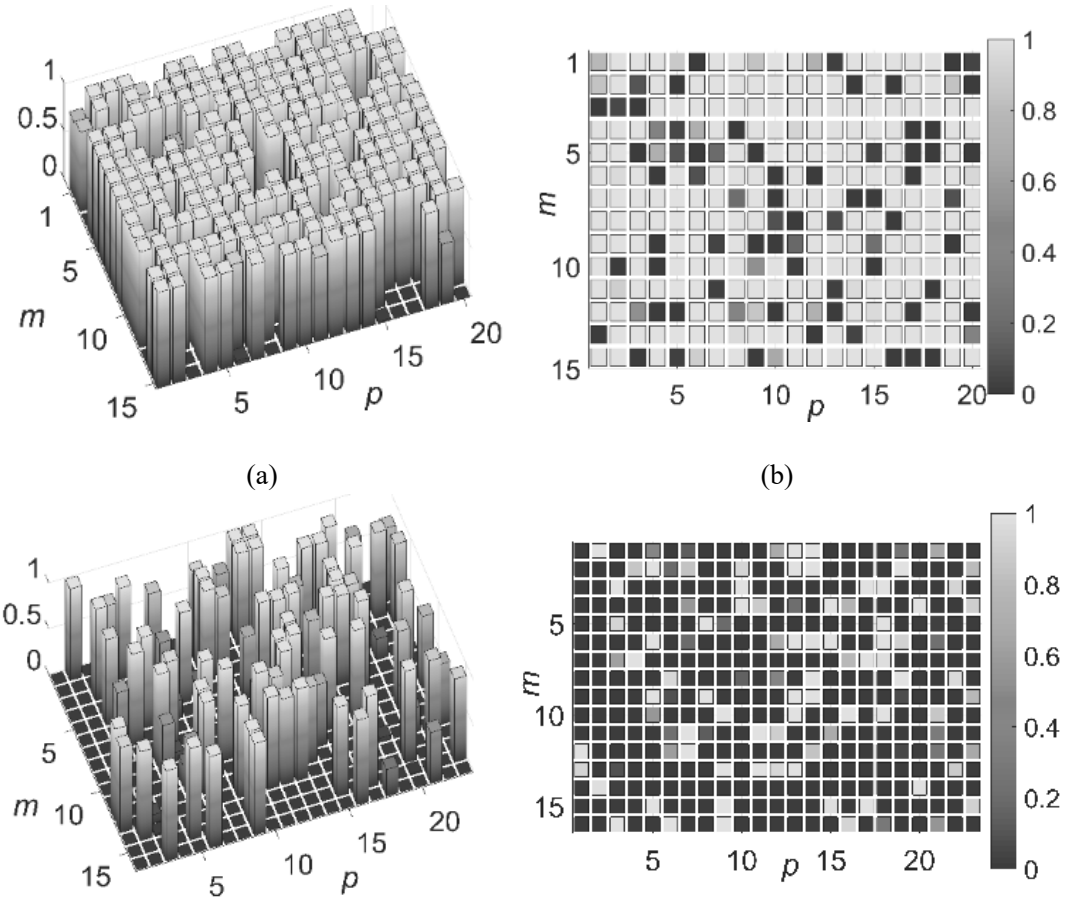
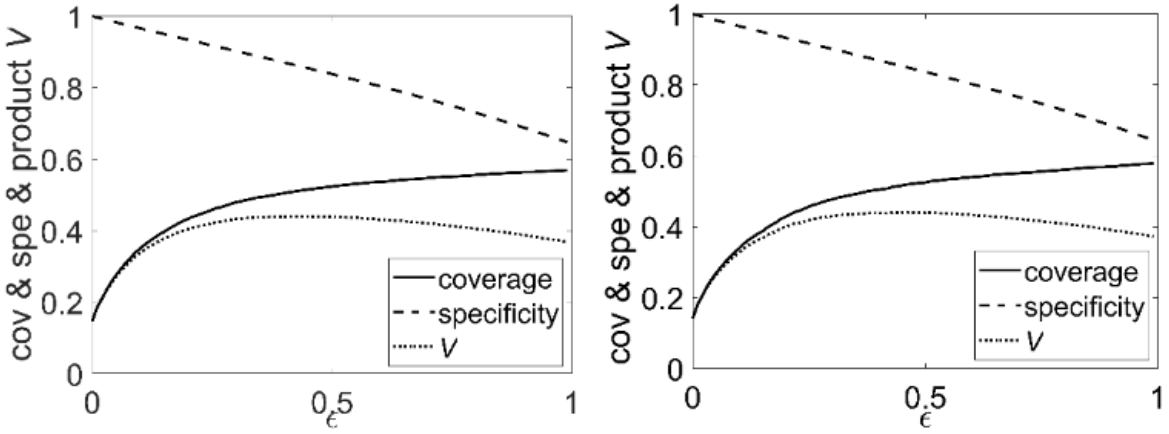


Fig. 3.11. Optimal relation G with R being optimized. (a) and (b) are the 3-D and 2-D view of G for the *s*-*t* factorization; (c) and (d) are the results for the *t*-*s* factorization.

Granular relational factorization:

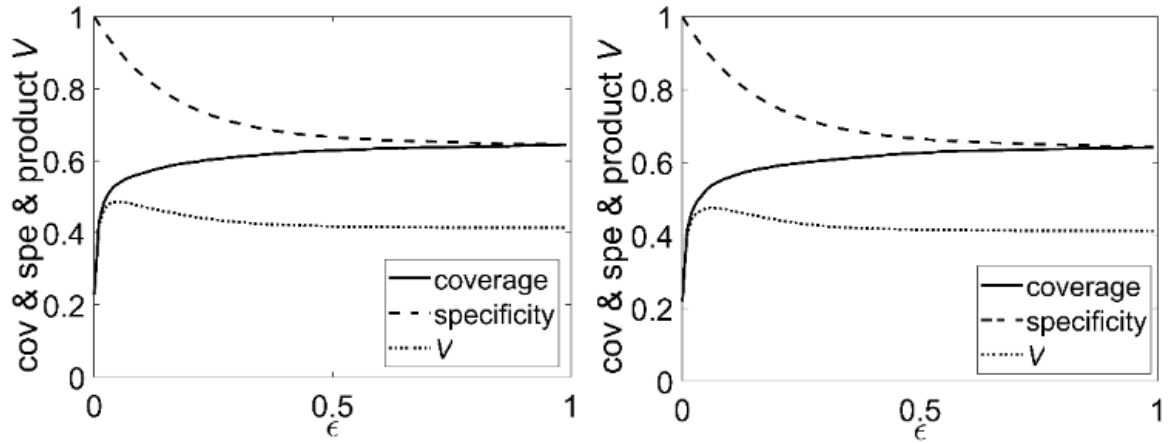
For the single-level architecture, the granular augmentation of the factorization results leads to the granular model, and fuzzy relation R is realized by admitting some level of information granularity ε. This value is determined experimentally by maximizing the product of coverage and specificity. In the experiment, its optimal value for *s*-*t* factorization model is determined by maximizing the product $V(\varepsilon)$ for the training data; the obtained value $\varepsilon_{\text{opt}} = 0.4$; as illustrated in Fig. 3.12. The corresponding values of V are $V_{\text{train}} = 0.4503$; $V_{\text{test}} = 0.4496$. The results for the

t-s factorization are included in Fig. 3.13. In this case $\varepsilon_{\text{opt}} = 0.09$, while $V_{\text{train}} = 0.485$; $V_{\text{test}} = 0.463$.



(a) (b)

Fig. 3.12. Granular augmentation of the *s-t* factorization through the maximization of the product of coverage and specificity V. (a) training data, (b) testing data

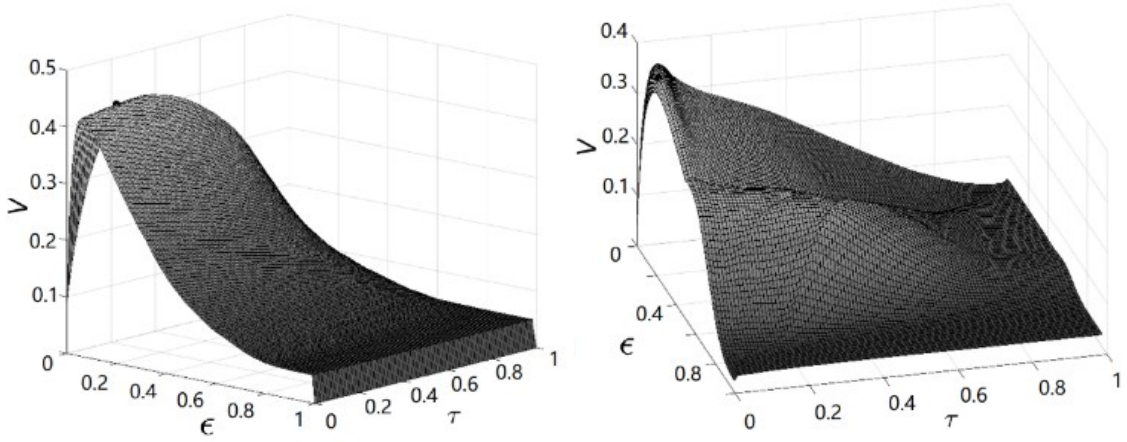


(a) (b)

Fig. 3.13. *t-s* factorization: coverage, specificity and V regarded as functions of ε. (a) training data, (b) testing data.

As for the NMF algorithm, we also use granular factorization based on the selected value of $p = 19$ (see Fig. 3.9). The optimal level of information granularity is $\varepsilon = 0.34$, $V_{train} = 0.355$, and $V_{test} = 0.356$.

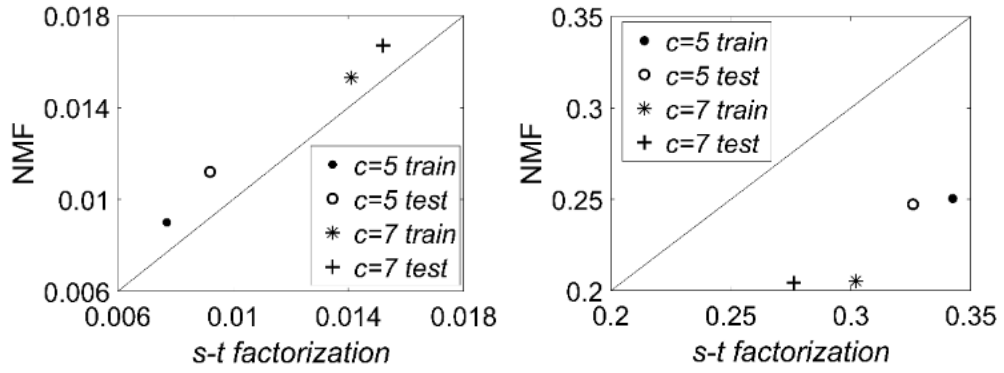
For the two-level factorization, we admit information granularity distributed across the entries of R and G. The optimization is completed for the pairs of values of ε and τ. The obtained results of V are contained in Fig. 3.14. The optimal level of information granularity is (a) $\varepsilon_{opt} = 0.03$; $\tau_{opt} = 0.19$, and the corresponding $V_{train} = 0.424$; $V_{test} = 0.416$. (b) $\varepsilon_{opt} = 0.18$; $\tau_{opt} = 0.01$, and the corresponding $V_{train} = 0.382$; $V_{test} = 0.378$ which is worse than the result of relational factorization. Both factorization methods perform better than the NMF algorithm with individual optimal parameters in terms of granular results. The maximal improvements are 30.06% and 16.85%, respectively when considering testing data.



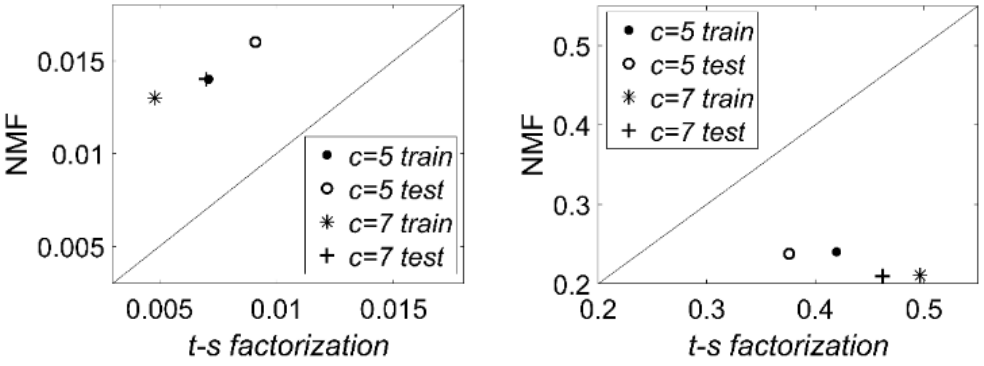
(a) (b)

Fig. 3.14. *V* as a function of ε and τ. (a) The *s-t* factorization; (b) The results for *t-s* factorization.

Repeat the experiments for different number of fuzzy sets, say 5 and 7 of Boston Housing data. The results are shown in Fig. 3.15.



(a) i. (a) ii.



(b) i. (b) ii.

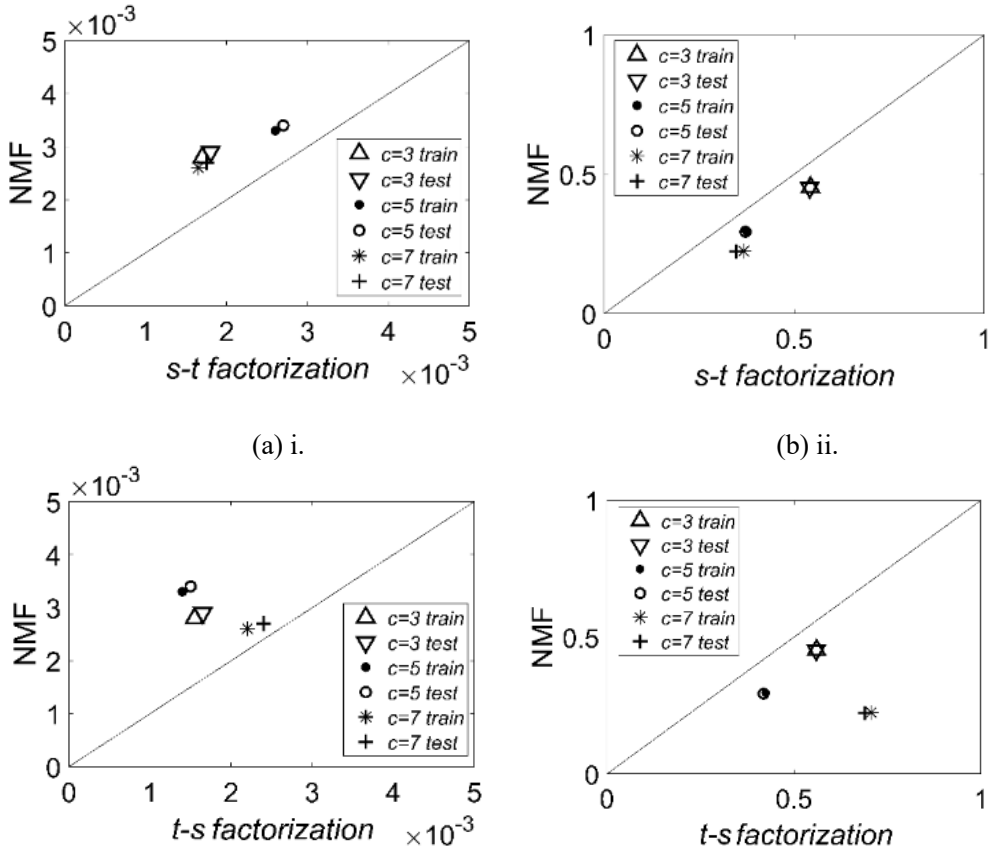
Fig. 3.15. Results for Boston Housing dataset. (a). Comparison of *s-t* factorization and NMF; (b). The results comparison of *t-s* factorization and NMF. (i) $\bar{Q}$; (ii) V.

In the above table we conclude that for *s-t* factorization, when $c = 5$ and $c = 7$, $\bar{Q}$ decreases by 14.4% and 7.8% in comparison with the results produced by the NMF, respectively. For *t-s* factorization, when $c = 5$ and $c = 7$, the decreases of $\bar{Q}$ are 49.3% and 63.1%, respectively.

Experiment 2:

Here we consider a number of publicly available data coming from the Machine Learning repository (https://archive.ics.uci.edu/ml/datasets.php). In other words the data we use for the experiments are generated by transforming real-world data into unit hypercubes with 3, 5 and 7 fuzzy sets. The results are reported in the following figures.

MAGIC Gamma Telescope DataSet (19,020 data with 10 input variables). Fig. 3.16 depicts the results of MAGIC Gamma Telescope Data Set with 3, 5 and 7 fuzzy sets.



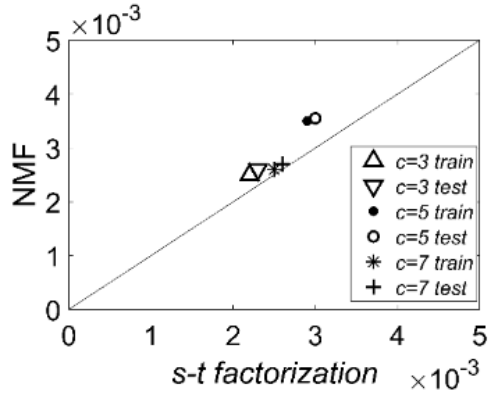
(a) i. (b) ii.

(b) i. (b) ii.

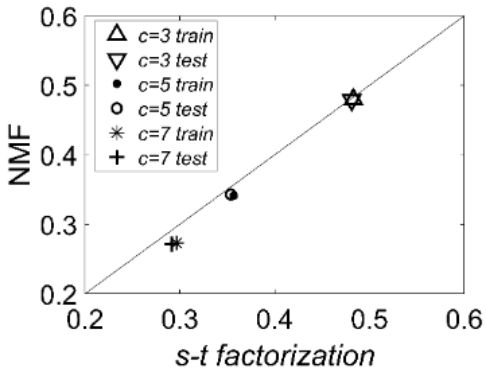
Fig. 3.16. Results for MAGIC Gamma Telescope dataset. (a). The results comparison of *s-t* factorization and NMF; (b) The results comparison of *t-s* factorization and NMF. (i) $\bar{Q}$; (ii) V.

For *s-t* factorization, when $c = 3$, $c = 5$ and $c = 7$, $\bar{Q}$ decreases by 39.3%, 21.2% and 36.5% in comparison with the performance results produced by the NMF, respectively. For *t-s* factorization, the decrease in the value of $\bar{Q}$ amounts to 44.6%, 57.6% and 15.4%, respectively.

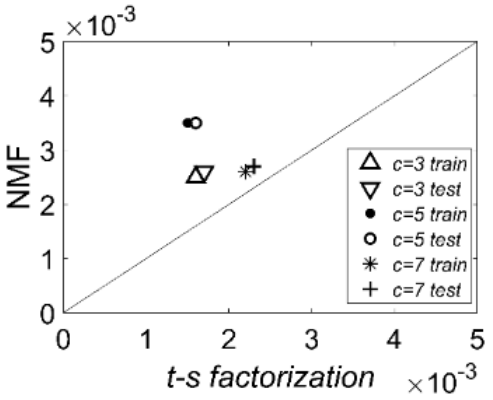
Wine Quality Data Set (4,898 data with 11 input variables). Fig. 3.17 includes the results for 3, 5 and 7 fuzzy sets.



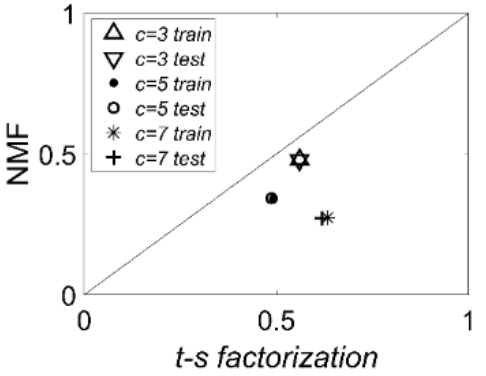
(a) i.



(a) ii.



(b) i.



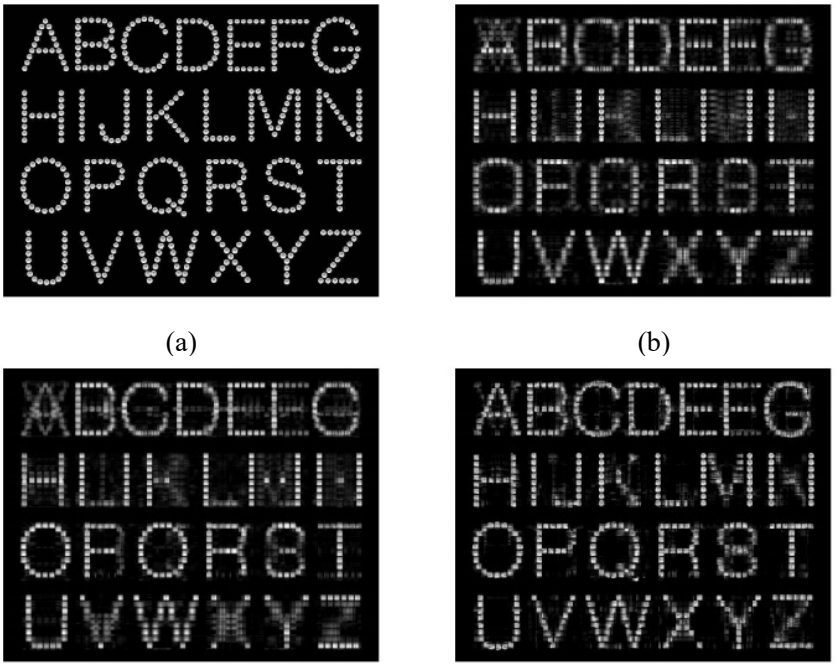
(b) ii.

Fig. 3.17. Results for Wine Quality Dataset. (a). The results of comparison of *s-t* factorization and NMF; (b). The results of comparison of *t-s* factorization and NMF. (i) $\bar{Q}$; (ii) V.

For *s-t* factorization, when $c = 3$, $c = 5$ and $c = 7$, $\bar{Q}$ decrease by 12.0%, 17.1% and 4.0% in comparison with the results produced by the NMF, respectively. For *t-s* factorization, $\bar{Q}$ decreases at the level 56.3%, 57.1% and 15.4% respectively.

Experiment 3:

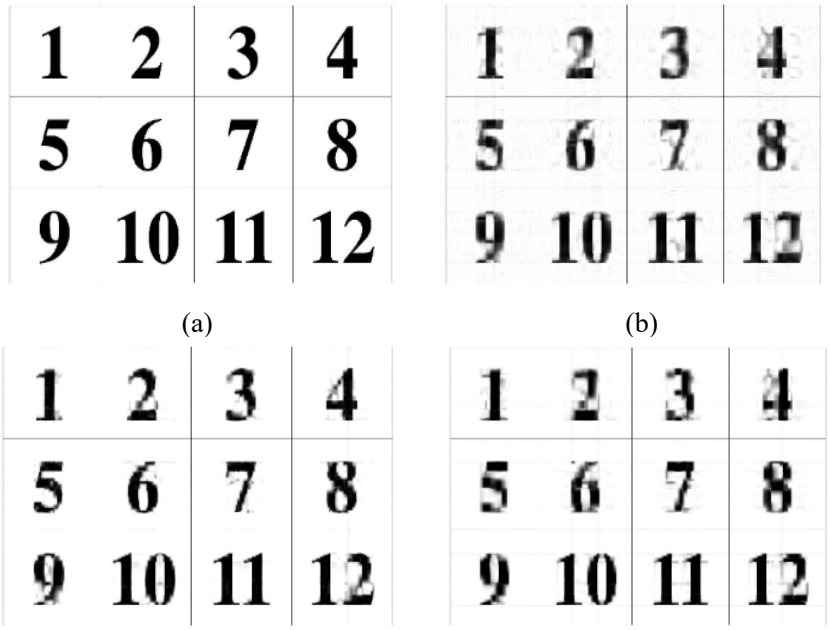
Here we use NMF, *s-t* factorization and *t-s* factorization to extract the image and compare the reconstruction image.



(a) (b)

(c) (d)

Fig. 3.18. The image of letters. (a) the original picture; (b) the reconstruction picture by NMF; the reconstruction picture: (c) *s*-*t* factorization; (d) *t*-*s* factorization.



(a) (b)

(c) (d)

Fig. 3.19. The image of numbers. (a) the original picture; (b) the reconstruction picture by NMF; the reconstruction picture: (c) *s*-*t* factorization; (d) *t*-*s* factorization.

In order to highlight the advantages of the developed model, we completed a comparative analysis with the MinMaxNMF algorithm [138] using the CBCL facial data. The data set consists of 2,429 faces and each image is composed of 19*19 pixels. The dimensionality of each datum is 361 (19*19 pixels). Then through relational factorization we reduce the dimensionality to 49. The performance index is the MSE. Table 3.1 contains the results of the comparative analysis. As is shown, the model has a significant improvement in terms of the reconstruction error.

Table 3.1. The comparison of results developed. (MinMaxNMF and relational factorization)

Algorithm	Error
MinMaxNMF	2.3e-3
t-s factorization	1.7e-4
s-t factorization	1.6e-4

In general, we can see that the performance for fuzzy relation factorization is better than the NMF. Especially for the *s-t* factorization, the error has decreased by 93%.

3.5 Conclusions

In this chapter, we have introduced a concept of the relational matrix factorization. The factorization process reduces the dimensionality of the original data by engaging mechanisms of logic processing involving generalized *and* and *or* processing completed with weighted (calibrated) *t*-norms and *t*-conorms. The optimization is realized by learning the logic expressions. The two-level structure reveals the essential ways of logic-oriented mapping of the original input variables and can find the essence of data by multiple logic. The reduced space of output variables is essential to the realization of a variety of models (say classifiers and rule-based models). The quantification of the quality of the relational factorization is completed through a construction of granular (interval-valued) relations which exhibit a direct impact on the ensuing models in the

sense the data of reduced dimensionality are made granular and as such are next being used in the construction of the models.

Chapter 4 Design of Fuzzy Rule-Based Models with Fuzzy Relational Factorization

As mentioned above, when faced with high-dimensional data, we can adapt the high-dimensional data by modifying and enhancing the model. Based on the relational factorization proposed in Chapter 3, we develop a new structure of fuzzy rule-based models with the fuzzy relation factorization. In this chapter, we briefly discuss the overall architecture and elaborate on the main functional modules and the development process. As illustrated in Fig. 4.1, the model is composed of three key functional modules, namely (i) encoding, (ii) fuzzy relational factorization, and (iii) a collection of rules along with the mapping process producing a numeric output of the model.

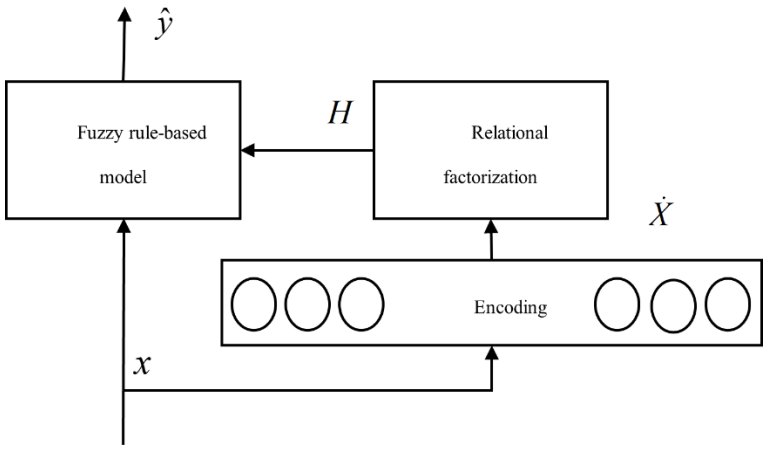


Fig. 4.1. Structure of fuzzy rule-based model based on fuzzy relation factorization.

Encoding Here input data are transformed (encoded) into their representations located in the unit hypercube. Each input variable defined is a collection of c information granules (fuzzy sets or intervals), with the use of which the variable is transformed to a collection of the corresponding membership degrees. This gives rise to a cn-dimensional vector positioned in the hypercube $[0,1]^{cn}$. In the case of categorical variables, 1-out-of c encoding is completed. The encoding

produces a nonlinear transformation of input data and forms a broad representation of the original data. The original data composed of $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$ build a data array $X = [x_1, x_2, \ldots, x_n]$ of dimensionality N by n, which gives rise to the array of encoded data $\dot{X} = [\dot{x}_1, \dot{x}_2, \ldots, \dot{x}_{cn}]$ of dimensionality N by cn.

Fuzzy relational factorization In the factorization process, $\dot{X}$ is decomposed (factorized) into data of lower dimensionality H and a relation (linkages) R, where $\dim(H) = p,\ p \leq cn$ and R is a p by cn matrix. This process effectively reduces the dimensionality of the original data. Formally, the factorization is expressed as $\dot{X} = H \circ R$, and the detailed process is discussed in Chapter 3.

Rule-based model The results of factorization conveyed by H are treated as vectors of activation levels of the rules of the model. The number of rules is p and the output of the model for any $\boldsymbol{x}$ is determined in a standard way as a weighted sum of $\boldsymbol{h}$ and local functions $L_i(\boldsymbol{x}; \boldsymbol{a}_i)$, $i = 1,2, \ldots, p$, say $\hat{y} = \sum_{i=1}^{p} h_i L_i(\boldsymbol{x}; \boldsymbol{a}_i)$. In the successive sections, we introduce step by step the details of the processing.

4.1 Encoding of input variables

The original input $\boldsymbol{x}$ is encoded with the aid of information granules defined for each variable. The essence of encoding is to represent any coordinate (variable) x_i in $\boldsymbol{x}$ through the corresponding collection of information granules (reference information granules) defined in the *i*-th variable. Formally speaking, assuming that for each variable there are *c* information granules, the *n*-dimensional vector $\boldsymbol{x}$ in R^n gives rise to a *cn*-dimensional vector of membership grades, forming a vector $\dot{\boldsymbol{x}}$ located in $[0,1]^{cn}$. The representation is sparse with a significant number of entries of $\dot{\boldsymbol{x}}$ equal to zero.

The reference information granules act as modules transforming original data in a nonlinear way. There are a number of possible ways of forming information granules. In this study, we consider several options.

(i) uniformly distributed triangular fuzzy sets with 1/2 overlap between two adjacent fuzzy sets. This alternative is straightforward not requiring any optimization overhead. The result of encoding is a vector with entries in [0,1], where up to two successive entries are nonzero and their values sum up to 1.

(ii) uniformly distributed intervals. The intervals form a partition of the space. The encoding returns a vector with a single entry equal to 1 and all others are set to 0.

(iii) equal probability intervals. The intervals are of unequal length – the lengths are selected in such way that each interval associates with the distribution of data; thus, there is the same probability of occurrence $(1/c)$ of each interval.

If some variable are categorical and take on c' values, the encoding results in one hot encoding, viz. a vector with c' coordinates with only one entry being equal to 1.

Once the encoding has been completed, the encoded data $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$ form a fuzzy relation $\dot{X}$ of dimensionality N by cn.

4.2 Relation factorization

Relational factorization is a logic-oriented way, see Chapter 3. The striking feature is in the realization of the underlying idea which exploits fuzzy relational calculus and as a consequence produces interpretable logic expressions. In turn they are used in conditions of the rules. Given data $\dot{X}$ produced by the encoding module, we have $\dot{X} = [0,1]^{cn}$. The factorization process decomposes $\dot{X}$ into two fuzzy relations H and R, see Section 3.1.

The matrix of relationships among original and reduced space is interpretable and gives rise to the logic expression describing new variables in H as a logic combination of input variables $\dot{X}$. Consider the j-th column of R. Its entries identify contributions of the input variables to the j-th variable used next as a condition of the rule. In light of the composition operator, we have two interpretations depending on the composition operator being used:

For *s-t* composition: the j-th variable is an *or* combination of weighted input variables, where the weights are located in the j-th column of R. The values of R close to 1 indicate essential contributions of the corresponding input variable. The values of R that are assumed close to 0 show a limited contribution of the input variable.

For *t-s* composition: the j-th variable is an *and* combination of weighted input variables, however a high relevance (contribution) of the input variable is noticeable for the entries of R assuming values close to 0.

The detailed calculations completed for selected *t*-norms and *t*-conorms are covered in Chapter 3. To sum up, the factorization returns a set of data in the space of lower dimensionality (p) and the fuzzy relation of linkages among the variables in the original and reduced space.

Once the factorization procedure has been completed, which results in the fuzzy relation R, for any new testing input data $\dot{\boldsymbol{x}}_{test}$ with $\dim(\dot{\boldsymbol{x}}_{test}) = cn$, we determine the corresponding $\boldsymbol{h}_{test}$ by minimizing the distance $\|\dot{\boldsymbol{x}}_{test} - \boldsymbol{h}_{test} \circ R\|^2$.

4.3 Construction of rule-based model

Consider the rule-based model with the structure

$$-\text{if } \boldsymbol{x} \text{ is } h_i \text{ then } y = L_i(\boldsymbol{x}) \tag{4.1}$$

The output for any given $\boldsymbol{x}$ is taken as a weighted sum of local functions in the following form

$$\hat{y} = \sum_{i=1}^{p} h_i L_i(\boldsymbol{x}) \tag{4.2}$$

where $L_i(\boldsymbol{x})$ is a local (usually linear) function. The activation levels of the local functions are organized in the vector form $\boldsymbol{h} = [h_1, h_2, \ldots, h_p]$, is computed on the basis of $\boldsymbol{x}$ that is encoded and afterwards obtained by the factorization process. A certain generalization of (4.2) is involved by admitting a certain transformation of the activation levels, namely, where $i = 1,2,\ldots,p$.

$$\hat{y} = \sum_{i=1}^{p} \Phi(h_i) L_i(\boldsymbol{x}) \tag{4.3}$$

Alone this line, the three alternatives of Φ are considered:

Method 1: identity function $\Phi(\boldsymbol{h}) = \boldsymbol{h}$.

Method 2: normalized version of $\boldsymbol{h}$, here $\Phi(\boldsymbol{h})$ is a vector whose maximal entry is equal to 1; the entries are specified as $[h_1/h^*, h_2/h^*, \ldots, h_p/h^*]$, where $h^* = \sum_{i=1}^{p} h_i$.

Method 3: similar to (Method 2) except that the maximal value of $\boldsymbol{h}$ is retained and all others are set to zero. Say, i_0 is the entry of $\boldsymbol{h}$, where the highest value is observed with $i_0 = \text{argmax}_i(h_i)$. $\Phi(\boldsymbol{h})$ is a vector in the form [0 0…0 1 0…0], where 1 occurs at the i_0^{th} entry of this vector.

Let us look in detail at the learning of the local functions. The learning is completed in the supervised model on the basis of the training data already used to complete relational factorization.

Denote by $\boldsymbol{h}_k$ the k^{th} row of H and subsequently $\Phi(\boldsymbol{h}_k)$. For the local functions, their parameters are estimated by minimizing the root mean square error.

The local functions are considered to be linear functions, (2.9) and constants $L_i(\boldsymbol{x}) = b_{i0}$. In both cases, the optimization problem is solved in an analogous manner. The optimal parameters for these two situations are determined as follows:

-linear function:

$$\boldsymbol{a}_i = (F^T F)^{-1} F^T \boldsymbol{target} \tag{4.4}$$

where

$$F = \begin{bmatrix} f_1^T(\boldsymbol{x}_1) & f_2^T(\boldsymbol{x}_1) & \cdots & f_p^T(\boldsymbol{x}_1) \\ f_1^T(\boldsymbol{x}_2) & f_2^T(\boldsymbol{x}_2) & \cdots & f_p^T(\boldsymbol{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1^T(\boldsymbol{x}_N) & f_2^T(\boldsymbol{x}_N) & \cdots & f_p^T(\boldsymbol{x}_N) \end{bmatrix};$$

$$\boldsymbol{f}_i(\boldsymbol{x}_k) = \begin{bmatrix} \Phi(h_{ki}) \\ \Phi(h_{ki})x_1 \\ \vdots \\ \Phi(h_{ki})x_n \end{bmatrix} \boldsymbol{a} = \begin{bmatrix} \boldsymbol{a}_1 \\ \boldsymbol{a}_2 \\ \vdots \\ \boldsymbol{a}_p \end{bmatrix}$$

-constant function:

$$\boldsymbol{b} = (\Phi^T \Phi)^{-1} \Phi^T \boldsymbol{target} \tag{4.5}$$

where

$$\Phi = \begin{bmatrix} \Phi(h_{11}) & \Phi(h_{12}) & \cdots & \Phi(h_{1p}) \\ \Phi(h_{21}) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \Phi(h_{N1}) & \cdots & \cdots & \Phi(h_{Np}) \end{bmatrix}$$

For comparison, we use the TS (Takagi-Sugeno) model designed in the standard way. The condition part is built with the use of the (fuzzy *c*-means) FCM with the fuzzification coefficient m being 2. Through the FCM clustering algorithm in [59], we obtain the membership grade A_i and prototypes $(\boldsymbol{v}_i, w_i)$, $i = 1,2,\ldots,p$. The local linear models forming the conclusion are in the form (2.9) or the constant model in the form $L_i(\boldsymbol{x}) = b_{i0}$ whose parameters are estimated by minimizing the Least Squares Error (*LSE*) criterion.

In addition, when it comes to the testing data transferred to the rule-based model, the structure of each rule comes from the above process, such as the selection of fuzzification coefficient and the number of clusters as well as the optimization of parameters in local models or the prototypes of output space. That is to say, we input testing data to the obtained rule-based model, generating the testing model output. Then the performance index Q is also considered as an evaluation criterion.

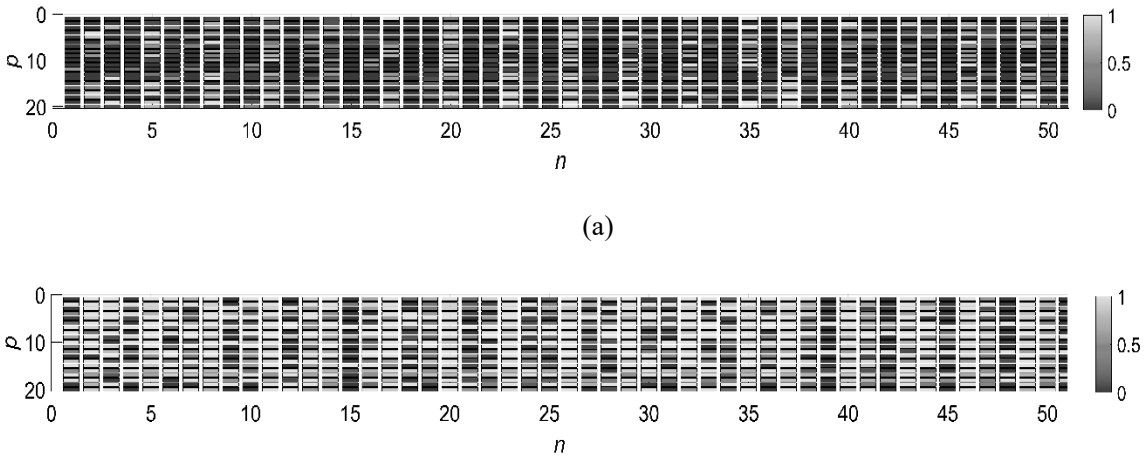
4.4 An illustrative example

In this section, we elaborate on the detailed design process by using the dataset YearPredictionMSD dataset (https://archive.ics.uci.edu/ml/datasets/yearpredictionmsd). It consists of 51,5345 data with 89 input variables and a single output assuming values in range $[1922, 2011]$. The experiments are carried out in a 10-fold cross validation mode.

Encoding of data The input variables are encoded with the aid of information granules as described in Section 3. The number of granules for each variable is three, which in total yields 267 $(89 * 3)$ input variables to be considered in the factorization procedure.

Fuzzy relational factorization The optimization problem completed here invokes the ADAptive moment estimation (ADAM) version of the gradient-based algorithm [113]. The number of iterations is set to 60, which is enough to reach the convergence of the method. We consider both the *t*-*s* and *s*-*t* compositions with the *t*-norm selected as product and *t*-conorms specified as the probabilistic sum.

We select the dimensionality of the reduced space obtained by the factorization to be $p = 20$. The obtained fuzzy relations (first 50 of 267 columns) are exposed in Fig. 4.2.



(a)

(b)

Fig. 4.2. Heatmaps of the fuzzy relation (weight matrix R). (a) *s*-*t* factorization; (b) *t*-*s* factorization.

From Table 4.1 we can see that the linear model has significant improvement compared with the constant model. The performance index of Method 1 is similar the Method 2 but better than Method 3.

Table 4.1. Performance index of the rule-based model obtained for training and testing data when $p = 20$.

Triangular membership	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:10.143±0.001 Q_{test}: 10.149±0.091	Q_{train}:10.143±0.001 Q_{test}: 10.149±0.091	Q_{train}:10.249±0.003 Q_{test}: 10.271±0.105	Q_{train}:9.246±0.006 Q_{test}: 9.536±0.026	Q_{train}:9.246±0.006 Q_{test}: 9.536±0.026	Q_{train}:9.259±0.009 Q_{test}: 9.449±0.064
t-s factorization	Q_{train}:10.197±0.003 Q_{test}: 10.215±0.109	Q_{train}:10.197±0.003 Q_{test}: 10.215±0.109	Q_{train}:10.359±0.007 Q_{test}: 10.376±0.063	Q_{train}:9.207±0.001 Q_{test}: 9.632±0.116	Q_{train}:9.207±0.001 Q_{test}: 9.632±0.116	Q_{train}: 9.530±0.005 Q_{test}: 9.830±0.050

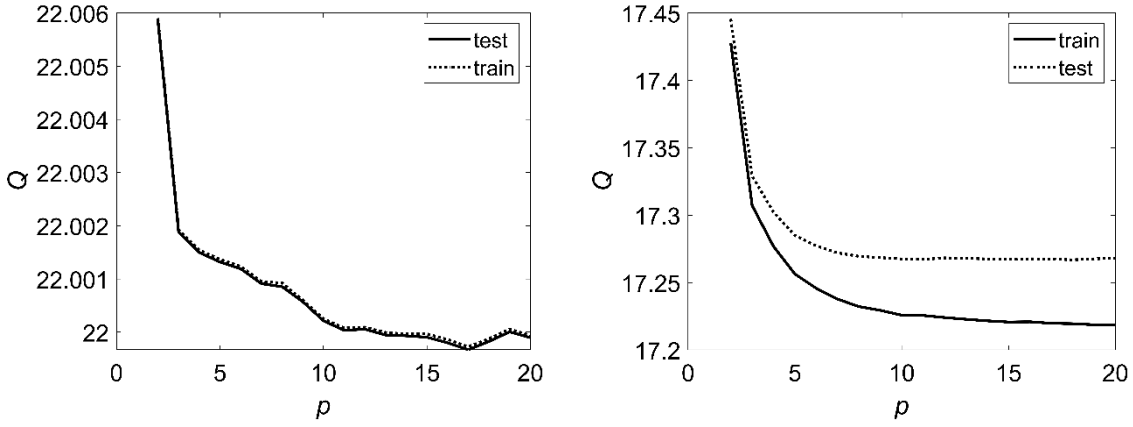
Uniform distribution	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:10.298±0.002 Q_{test}: 10.384±0.050	Q_{train}:10.298±0.002 Q_{test}: 10.384±0.050	Q_{train}:10.316±0.002 Q_{test}: 10.367±0.038	Q_{train}:9.433±0.002 Q_{test}: 9.723±0.074	Q_{train}:9.433±0.002 Q_{test}: 9.723±0.074	Q_{train}: 9.449±0.002 Q_{test}: 9.729±0.076
t-s factorization	Q_{train}:10.374±0.003 Q_{test}: 10.439±0.053	Q_{train}:10.374±0.003 Q_{test}: 10.439±0.053	Q_{train}:10.493±0.004 Q_{test}: 10.563±0.035	Q_{train}:9.695±0.003 Q_{test}: 10.185±0.269	Q_{train}:9.695±0.003 Q_{test}: 10.185±0.269	Q_{train}: 9.679±0.003 Q_{test}: 10.041±0.104

Equal probability	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:10.456±0.005 Q_{test}: 10.489±0.105	Q_{train}:10.456±0.005 Q_{test}: 10.489±0.105	Q_{train}:10.506±0.003 Q_{test}: 10.536±0.094	Q_{train}:9.381±0.008 Q_{test}: 9.657±0.073	Q_{train}:9.381±0.008 Q_{test}: 9.657±0.073	Q_{train}:9.471±0.008 Q_{test}: 9.733±0.112
t-s factorization	Q_{train}:10.474±0.003 Q_{test}: 10.477±0.096	Q_{train}:10.474±0.003 Q_{test}: 10.477±0.096	Q_{train}:10.532±0.005 Q_{test}: 10.560±0.096	Q_{train}:9.413±0.004 Q_{test}: 9.645±0.135	Q_{train}:9.413±0.004 Q_{test}: 9.645±0.135	Q_{train}:9.593±0.003 Q_{test}: 9.846±0.127

For the obtained *s-t* model, most of inputs associate with the entries of the relation are equal to 0, and only a few entries of R exhibit values close to 1. For instance, when inspecting the first row, see Fig. 4.3(a), only 9 weights are close to 1, while others assume values close to zero. In contrast for the *t-s* model, the situation is opposite: the original inputs contributing to the formation of the results of factorization are those, whose corresponding entries of R assume values close to zero. In both cases, we note that only a fraction of inputs contributes to the results of factorization. For example, for the *s-t* factorization, the columns 5, 8, 11, 20 and 23 play a key role. The 9, 12, 15 and 18 columns are more important.

Optimization of the rules in the reduced space of fuzzy factorization. The rules are considered in the form (2.5). Both the constant and linear conclusions are considered. Also the three transformation functions Φ are involved. The results are collected in Table 4.1.

As a comparison, we introduce the TS model with the same dataset. The results shown in Fig. 4.3 are the performance index V as a function of number of clusters. In Table 4.2, we depict the specific results with $p = 20$.



(a) (b)

Fig. 4.3. Values of performance index V for TS model obtained in successive values of p. (a) constant local functions; (b) linear local functions.

Table 4.2. Performance index of the TS model obtained for training and testing data. $p = 20$.

	p	Constant model	Linear Model
TS model	20	Q_{train}:22.000±0.001 Q_{test}: 22.001±0.004	Q_{train}:17.226±4.3e-04 Q_{test}: 17.670±0.035

Compared with the TS model, we can see that the error Q of the rule-based model involving factorization is better than the one obtained for the TS model. For the constant model, the constant function is considered for the conclusion part when $p = 20$. The performance of our new model improves by 53.9% regarding the testing result. For this linear model, the improvement is 46.0%.

4.5 Experimental studies

In the subsequent experiments, we use a number of data Parkinsons Telemonitoring, Concrete Compressive Strength, coming from UCI machine learning datasets (https://archive.ics.uci.edu/ml/index.php) and LightGBM's regression examples from Kaggle datasets (https://www.kaggle.com). In the reported results, we use $p = 10$ as an example to show the improvement of our model.

Table 4.3. The list of datasets.

Dataset	Size of Data	Size after encoding (c=3)	Range of output
Parkinsons Telemonitoring	5875×17	5875×49	[7, 54.99]
Concrete Compressive Strength	1030×9	1030×25	[2.33, 82.6]
LightGBMs regression example	7000×29	7000×85	[0.489, 4.316]
Appliances energy prediction	19735×25	19735×73	[10, 1080]
Impact of News on the Share closing value (Apple)	2517×9	2517×25	[-0.997, 0.999]

Parkinsons Telemonitoring (48 inputs with 1 output (total_UPDRS) after encoding)

(https://archive.ics.uci.edu/ml/datasets/Parkinsons+Telemonitoring)

Table 4.4. Performance index of the rule-based model for training and testing data when $p = 10$.

Triangular membership	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:20.314±0.065 Q_{test}: 20.591±4.288	Q_{train}: 20.314±0.065 Q_{test}: 20.591±4.288	Q_{train}:20.731±0.040 Q_{test}: 20.758±4.481	Q_{train}:8.886±0046 Q_{test}: 9.646±3.380	Q_{train}:8.886±0046 Q_{test}: 9.646±3.380	Q_{train}:9.224±0.154 Q_{test}: 10.134±5.104
t-s factorization	Q_{train}:20.251±0.056 Q_{test}: 20.609±4.848	Q_{train}:20.251±0.056 Q_{test}: 20.609±4.848	Q_{train}:21.151±0.038 Q_{test}: 21.305±3.905	Q_{train}:8.947±0.032 Q_{test}: 9.738±1.996	Q_{train}:8.947±0.032 Q_{test}: 9.738±1.996	Q_{train}:9.405±0.034 Q_{test}: 10.609±1.590

Uniform distribution	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:20.899±0.121 Q_{test}: 20.973±5.414	Q_{train}:20.899±0.121 Q_{test}: 20.973±5.414	Q_{train}:21.215±0.137 Q_{test}: 21.225±6.215	Q_{train}:9.447±0.028 Q_{test}: 9.600±4.119	Q_{train}:9.447±0.028 Q_{test}: 9.600±4.119	Q_{train}:9.706±0.036 Q_{test}: 9.800±4.635
t-s factorization	Q_{train}:20.712±0.085 Q_{test}: 20.734±5.201	Q_{train}:20.712±0.085 Q_{test}: 20.734±5.201	Q_{train}:21.368±0.105 Q_{test}: 21.209±4.456	Q_{train}: 9.091±0.028 Q_{test}: 10.419±4.272	Q_{train}: 9.091±0.028 Q_{test}: 10.419±4.272	Q_{train}:9.745±0.067 Q_{test}: 10.890±5.168

Equal probability	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:20.659±0.146 Q_{test}: 20.704±5.323	Q_{train}:20.659±0.146 Q_{test}: 20.704±5.323	Q_{train}:20.905±0.166 Q_{test}: 20.947±5.974	Q_{train}:9.418±0.021 Q_{test}: 10.772±4.889	Q_{train}:9.418±0.021 Q_{test}: 10.772±4.889	Q_{train}:9.897±0.054 Q_{test}: 10.145±4.103
t-s factorization	Q_{train}:20.737±0.114 Q_{test}: 20.749±5.096	Q_{train}:20.737±0.114 Q_{test}: 20.749±5.096	Q_{train}:21.415±0.138 Q_{test}: 21.842±5.530	Q_{train}:9.155±0.038 Q_{test}: 10.842±4.936	Q_{train}:9.155±0.038 Q_{test}: 10.842±4.936	Q_{train}:9.948±0.040 Q_{test}: 10.182±4.275

TS model	Constant model	Linear Model
	Q_{train}:21.637±0.092 Q_{test}: 22.791±7.583	Q_{train}:15.424±0.019 Q_{test}: 17.270±2.012

Concrete Compressive Strength (24 inputs with a single output)

(https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength)

Table 4.5. Performance index of the rule-based model for training and testing data when $p = 10$.

Triangular membership	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:12.193±0.307 Q_{test}: 12.521±0.815	Q_{train}:12.193±0.307 Q_{test}: 12.521±0.815	Q_{train}:13.006±0.202 Q_{test}: 13.469±0.787	Q_{train}:6.595±0.051 Q_{test}: 7.742±0.593	Q_{train}:6.595±0.051 Q_{test}: 7.742±0.593	Q_{train}:6.649±0.053 Q_{test}: 7.885±0.857
t-s factorization	Q_{train}:11.914±0.188 Q_{test}: 12.158±0.566	Q_{train}:11.914±0.188 Q_{test}: 12.158±0.566	Q_{train}:13.623±0.359 Q_{test}: 13.943±0.547	Q_{train}:6.518±0.044 Q_{test}: 7.546±0.560	Q_{train}:6.518±0.044 Q_{test}: 7.546±0.560	Q_{train}:6.980±0.200 Q_{test}: 7.821±0.390

Uniform distribution	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:13.971±0.066 Q_{test}: 14.192±0.488	Q_{train}:13.971±0.066 Q_{test}: 14.192±0.488	Q_{train}:14.764±0.043 Q_{test}: 15.274±0.714	Q_{train}:7.638±0.023 Q_{test}: 8.751±0.460	Q_{train}:7.638±0.023 Q_{test}: 8.751±0.460	Q_{train}:8.028±0.014 Q_{test}: 9.398±1.063
t-s factorization	Q_{train}:14.413±0.211 Q_{test}: 14.484±1.170	Q_{train}:14.413±0.211 Q_{test}: 14.484±1.170	Q_{train}:15.274±0.090 Q_{test}: 15.740±0.569	Q_{train}:7.953±0.080 Q_{test}: 9.293±1.405	Q_{train}:7.953±0.080 Q_{test}: 9.293±1.405	Q_{train}:8.372±0.076 Q_{test}: 9.908±2.172

Equal probability	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:11.041±0.750 Q_{test}: 11.200±0.789	Q_{train}:11.041±0.750 Q_{test}: 11.200±0.789	Q_{train}:12.966±0.285 Q_{test}: 13.315±0.786	Q_{train}:6.075±0.411 Q_{test}: 6.926±0.748	Q_{train}:6.075±0.411 Q_{test}: 6.926±0.748	Q_{train}:6.659±0.320 Q_{test}: 7.982±0.262
t-s factorization	Q_{train}:12.576±0.881 Q_{test}: 12.577±1.220	Q_{train}:12.576±0.881 Q_{test}: 12.577±1.220	Q_{train}:14.322±0.145 Q_{test}: 14.369±0.958	Q_{train}:6.493±0.082 Q_{test}: 7.814±0.752	Q_{train}:6.493±0.082 Q_{test}: 7.814±0.752	Q_{train}:7.444±0.032 Q_{test}: 9.261±0.978

	Constant model	Linear Model
TS model	Q_{train}:15.318±0.017 Q_{test}: 15.387±0.828	Q_{train}:8.229±0.052 Q_{test}: 9.067±0.847

LightGBM's regression examples (84 inputs with a single output)

(https://www.kaggle.com/cccccao/regression)

Table 4.6. Performance index of the rule-based model for training and testing data when $p = 10$.

Triangular membership	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:0.281±7.5e-05 Q_{test}: 0.281±8.6e-05	Q_{train}:0.281±7.5e-05 Q_{test}: 0.281±8.6e-05	Q_{train}:0.295±2.1e-05 Q_{test}: 0.296±1.5e-04	Q_{train}:0.121±1.8e-06 Q_{test}: 0.123±1.1e-04	Q_{train}:0.121±1.8e-06 Q_{test}: 0.123±1.1e-04	Q_{train}:0.119±1.3e-06 Q_{test}: 0.123±1.1e-04
t-s factorization	Q_{train}:0.305±3.2e-05 Q_{test}: 0.306±1.7e-04	Q_{train}:0.305±3.2e-05 Q_{test}: 0.306±1.7e-04	Q_{train}:0.303±3.7e-05 Q_{test}: 0.305±2.4e-04	Q_{train}:0.124±1.2e-06 Q_{test}: 0.126±1.1e-04	Q_{train}:0.124±1.2e-06 Q_{test}: 0.126±1.1e-04	Q_{train}:0.123±1.2e-06 Q_{test}: 0.125±1.1e-04

Uniform distribution	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:0.315±5.3e-06 Q_{test}: 0.316±2.8e-04	Q_{train}:0.315±5.3e-06 Q_{test}: 0.316±2.8e-04	Q_{train}:0.315±4.1e-06 Q_{test}: 0.316±3.1e-04	Q_{train}:0.124±2.1e-06 Q_{test}: 0.126±1.6e-04	Q_{train}:0.124±2.1e-06 Q_{test}: 0.126±1.6e-04	Q_{train}:0.122±1.9e-06 Q_{test}: 0.126±1.5e-04
t-s factorization	Q_{train}:0.314±5.7e-06 Q_{test}: 0.314±2.8e-04	Q_{train}:0.314±5.7e-06 Q_{test}: 0.314±2.8e-04	Q_{train}:0.315±3.4e-06 Q_{test}: 0.315±2.7e-04	Q_{train}:0.125±2.2e-06 Q_{test}: 0.126±1.7e-04	Q_{train}:0.125±2.2e-06 Q_{test}: 0.126±1.7e-04	Q_{train}:0.122±2.1e-06 Q_{test}: 0.127±1.7e-04

Equal probability	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:0.238±9.2e-06 Q_{test}: 0.239±4.0e-04	Q_{train}:0.238±9.2e-06 Q_{test}: 0.239±4.0e-04	Q_{train}:0.254±1.8e-05 Q_{test}: 0.254±4.5e-04	Q_{train}:0.123±1.4e-06 Q_{test}: 0.125±9.5e-05	Q_{train}:0.123±1.4e-06 Q_{test}: 0.125±9.5e-05	Q_{train}:0.121±1.6e-06 Q_{test}: 0.125±1.0e-04
t-s factorization	Q_{train}:0.248±5.7e-06 Q_{test}: 0.248±3.1e-04	Q_{train}:0.248±5.7e-06 Q_{test}: 0.248±3.1e-04	Q_{train}:0.276±8.9e-06 Q_{test}: 0.277±3.8e-04	Q_{train}:0.116±2.4e-06 Q_{test}: 0.125±1.3e-04	Q_{train}:0.116±2.4e-06 Q_{test}: 0.125±1.3e-04	Q_{train}:0.119±2.0e-06 Q_{test}: 0.128±1.3e-04

	Constant model	Linear Model
TS model	Q_{train}:0.316±5.5e-06 Q_{test}: 0.326±4.9e-04	Q_{train}:0.176±1.5e-06 Q_{test}: 0.184±1.3e-04

Appliance's energy prediction (72 inputs with a single output (Appliances, energy use in Wh))

(https://archive.ics.uci.edu/ml/datasets/Appliances+energy+prediction)

Table 4.7. Performance index of the rule-based model for training and testing data when $p = 10$.

Triangular membership	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:100.468±0.29 Q_{test}:101.562±27.43	Q_{train}:100.468±0.29 Q_{test}: 101.562±27.43	Q_{train}:100.838±0.31 Q_{test}:100.632±26.15	Q_{train}:91.776±0.32 Q_{test}: 92.153±23.81	Q_{train}:91.776±0.32 Q_{test}: 92.153±23.81	Q_{train}:90.545±0.29 Q_{test}: 91.234±22.83
t-s factorization	Q_{train}:100.403±0.27 Q_{test}:101.339±27.61	Q_{train}:100.403±0.27 Q_{test}: 101.339±27.61	Q_{train}:100.069±0.32 Q_{test}:101.022±28.67	Q_{train}:94.214±0.30 Q_{test}: 94.281±27.01	Q_{train}:94.214±0.30 Q_{test}: 94.281±27.01	Q_{train}:93.088±0.29 Q_{test}: 93.332±26.71

Uniform distribution	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:100.780±0.13 Q_{test}:101.900±10.51	Q_{train}:100.780±0.13 Q_{test}: 101.900±10.51	Q_{train}:100.203±0.11 Q_{test}:101.177±10.34	Q_{train}:91.977±0.19 Q_{test}: 92.415±9.26	Q_{train}:91.977±0.19 Q_{test}: 92.415±9.26	Q_{train}:90.950±0.20 Q_{test}: 91.711±9.14
t-s factorization	Q_{train}:100.699±0.21 Q_{test}:101.795±10.20	Q_{train}:100.699±0.21 Q_{test}: 101.795±10.20	Q_{train}:100.597±0.17 Q_{test}:101.634±10.31	Q_{train}:94.037±0.10 Q_{test}: 94.197±9.22	Q_{train}:94.037±0.10 Q_{test}: 94.197±9.22	Q_{train}:92.709±0.09 Q_{test}: 93.312±9.43

Equal probability	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:100.791±0.85 Q_{test}: 101.082±66.95	Q_{train}:100.791±0.85 Q_{test}:101.082±66.95	Q_{train}:100.954±0.782 Q_{test}: 101.234±66.49	Q_{train}:92.201±0.68 Q_{test}: 92.504±50.68	Q_{train}:92.201±0.68 Q_{test}: 92.504±50.68	Q_{train}:91.209±0.77 Q_{test}: 92.074±48.91
t-s factorization	Q_{train}:100.551±0.89 Q_{test}: 101.779±65.86	Q_{train}:100.551±0.89 Q_{test}:101.779±65.86	Q_{train}:100.054±0.76 Q_{test}: 101.345±65.81	Q_{train}:92.961±0.59 Q_{test}: 93.984±53.86	Q_{train}:92.961±0.59 Q_{test}: 93.984±53.86	Q_{train}:92.601±0.81 Q_{test}: 92.987±49.95

	Constant model	Linear Model
TS model	Q_{train}:126.295±1.05 Q_{test}: 135.306±30.496	Q_{train}:116.621±2.372 Q_{test}: 120.242±60.017

Impact of News on the Share closing value (Apple) (24 inputs with a single output)

(https://www.kaggle.com/BidecInnovations/stock-price-and-news-realted-to-it)

Table 4.8. Performance index of the rule-based model for training and testing data when $p = 10$.

Triangular membership	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:0.355±0.001 Q_{test}: 0.556±0.001	Q_{train}:0.355±0.001 Q_{test}: 0.556±0.001	Q_{train}:100.838±0.31 Q_{test}:100.632±26.15	Q_{train}:0.316±9.3e-05 Q_{test}: 0.319±3.9e-04	Q_{train}:0.316±9.3e-05 Q_{test}: 0.319±3.9e-04	Q_{train}:0.308±4.8e-05 Q_{test}: 0.314±5.9e-04
t-s factorization	Q_{train}:0.337±4.1e-04 Q_{test}: 0.344±9.9e-04	Q_{train}:0.337±4.1e-04 Q_{test}: 0.344±9.9e-04	Q_{train}:100.069±0.32 Q_{test}:101.022±28.67	Q_{train}:0.344±4.8e-05 Q_{test}: 0.347±4.1e-04	Q_{train}:0.344±4.8e-05 Q_{test}: 0.347±4.1e-04	Q_{train}:0.328±1.5e-05 Q_{test}: 0.331±5.7e-04

Uniform distribution	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:0.519±8.4e-05 Q_{test}: 0.529±7.1e-04	Q_{train}:0.519±8.4e-05 Q_{test}: 0.529±7.1e-04	Q_{train}:0.540±2.3e-04 Q_{test}: 0.549±7.6e-04	Q_{train}:0.311±1.2e-04 Q_{test}: 0.323±5.9e-04	Q_{train}:0.311±1.2e-04 Q_{test}: 0.323±5.9e-04	Q_{train}:0.311±1.2e-04 Q_{test}: 0.331±9.4e-04
t-s factorization	Q_{train}:0.526±3.3e-05 Q_{test}: 0.528±2.1e-04	Q_{train}:0.526±3.3e-05 Q_{test}: 0.528±2.1e-04	Q_{train}:0.593±4.8e-05 Q_{test}: 0.594±2.8e-04	Q_{train}:0.346±2.9e-06 Q_{test}: 0.349±1.9e-04	Q_{train}:0.346±2.9e-06 Q_{test}: 0.349±1.9e-04	Q_{train}:0.339±2.0e-05 Q_{test}: 0.350±2.7e-04

Equal probability	Constant			Linear		
	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
s-t factorization	Q_{train}:0.479±0.001 Q_{test}: 0.497±0.005	Q_{train}:0.479±0.001 Q_{test}: 0.497±0.005	Q_{train}:0.522±9.2e-04 Q_{test}: 0.538±0.002	Q_{train}:0.314±5.4e-05 Q_{test}: 0.315±5.2e-04	Q_{train}:0.314±5.4e-05 Q_{test}: 0.315±5.2e-04	Q_{train}:0.310±6.7e-05 Q_{test}: 0.319±1.9e-04
t-s factorization	Q_{train}:0.451±3.0e-04 Q_{test}: 0.452±9.4e-4	Q_{train}:0.451±3.0e-04 Q_{test}: 0.452±9.4e-4	Q_{train}:0.531±7.3e-04 Q_{test}: 0.538±0.001	Q_{train}:0.341±6.9e-06 Q_{test}: 0.342±6.1e-04	Q_{train}:0.341±6.9e-06 Q_{test}: 0.342±6.1e-04	Q_{train}:0.322±1.5e-05 Q_{test}: 0.328±5.9e-04

	Constant model	Linear Model
TS model	Q_{train}:0.625±6.3e-05 Q_{test}: 0.628±2.7e-04	Q_{train}:0.339±2.4e-05 Q_{test}: 0.363±1.4e-04

From the above tables, we conclude that the linear model is better than the constant model. Among the three different membership functions that we adopt, triangular membership has better performance than Uniform distribution and Equal probability; while the improvement of *s-t* factorization and *t-s* factorization ultimately depends on the different data and the chosen method, but in most cases, the *s-t* factorization performs better. For the parameter p, the increase of p will significantly reduce the error of the model.

Table 4.9 is the improvement between our new model and TS model for each data when $p = 10$.

Table 4.9. The improvement obtained for each dataset.

Data	Constant model	Linear model
Parkinsons Telemonitoring	25%	24%
Concrete Compressive Strength	21%	17%
LightGBMs regression	27%	33%
Appliance's energy prediction	26%	24%
Impact of News on the Share closing value (Apple)	45%	14%

Overall, for the constant model, our model is optimized by an average of 28.8%, and for the linear model, the optimization reaches an average of 22.4%.

4.6 Conclusions

In the study, the design process of rule-based models augmented with the mechanism of relational factorization is presented. The construction of the rules required to be completed in the presence of high-dimensional data is facilitated by the formation of the condition part as logic expressions formed by the factorization process. In the experiments, we adopted three encoding schemes and combined three different methods to compare the *s*-*t* and *t*-*s* models, and we designed experiments for the constant and linear models respectively. Finally, the feasibility of the model is confirmed by multiple sets of data.

Chapter 5 Distributed Rule-Based Models with Large Data

In the previous two chapters, we have shown the methods for dealing with high dimensional data by preprocessing the data (dimensionality reduction) and extracting the underlying relationships of the data, respectively. In this chapter, we report on the research on the fuzzy rule-based model with high-dimensional data and propose a new method based on the idea of ensemble learning. We develop a fuzzy rule-based model based on a distributed model to circumvent high-dimensional data, and the performance of the model is improved.

5.1 An architecture of the model and its underlying processing

When faced with big data problems, we usually divide it into two different problems, namely large-scale data problems and large-dimensional data problems. Given a large amount of data, we can usually take random sampling to extract a certain amount of data for analysis. The process of random sampling can ensure the integrity of the data to the greatest extent. For high-dimensional data, the problem becomes complicated due to the concentration effect. This is because in the case of high dimensions, the indicator of distance becomes no longer reliable. Since high-dimensional data is unavoidable in the current environment. In this chapter, we continue to discuss how to solve the problem of fuzzy rule-based models based on high-dimensional data.

The structure of the overall model is composed of a collection of rule-based models that are built on a basis of randomly selected subsets of data.

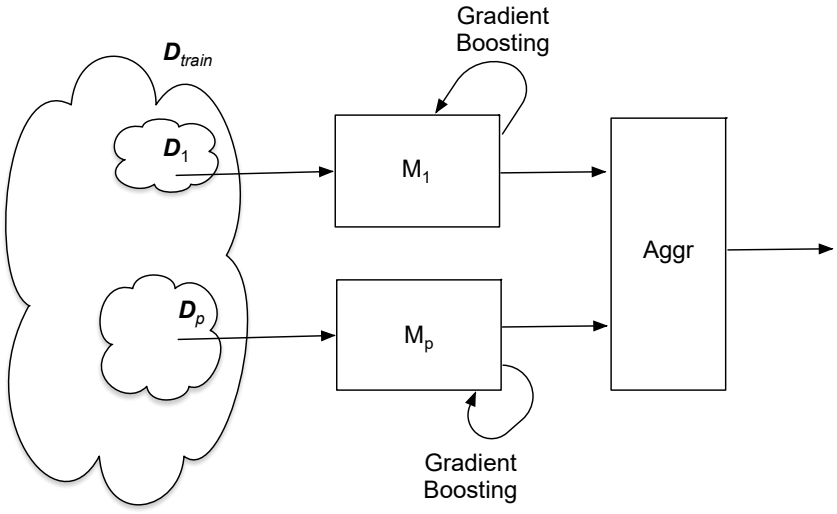


Fig. 5.1. Overall structure of the model and its functioning.

The main steps of processing completed by each module as outlined in Fig. 5.1 are carried out as follows:

(i) training of the rule-based models completed on a basis of randomly selected data. A subset of training data is composed of randomly selected data and randomly selected features.

(ii) for each subset of data formed above, the standard design process of the TS rule-based model is completed. First, the data are clustered which leads to the condition parts of the rules. Next the local linear functions forming the conclusions of the rules are optimized by minimizing a sum of squared errors.

(iii) the above design process is augmented by enhancing the performance of the models by engaging its gradient boosting.

(iv) finally, the results of the constructed models are aggregated by the aggregation module; here a linear weighted aggregation scheme is usually considered.

5.2 The design of main modules of the model

In what follows, we proceed in detail with an optimization associated with the architecture in Fig. 5.1. As usual, in the overall design process, the available data $\boldsymbol{D}$ are split into the training $\boldsymbol{D}_{train}$ and testing $\boldsymbol{D}_{test}$.

Generic rule-based model

The rules come in the standard format as (2.9). For the design of the rule-based model, the number of rules varies across the models. The parameters of the local functions are determined by minimizing the sum of squared errors

$$error = \frac{1}{N^*}\sum_{j=1}^{N^*}\Big(target_j - L_i\big(x_j\big)\Big)^2 \tag{5.1}$$

where the above sum is taken from the corresponding randomly selected data.

In total, we consider p rule-based models. An interesting question arises as to the usage of all features in $\boldsymbol{D}_{train}$ across the subsets of training data $\boldsymbol{D}_{train,j}, j = 1, 2, \dots, p$. Note that along with the data, we are also randomly picking up a subset of r out of n features. Thus, it is of interest to assess how many models (p) have to be built to involve all features in the construction of the aggregated model. The probability *prob* of this event (stating that each variable being chosen) is expressed as [139]

$$prob = r + (1-r)r + (1-r)^2 r + \cdots + (1-r)^p r = r\left(\frac{1-(1-r)^p}{1-(1-r)}\right) \tag{5.2}$$

If we require a certain level of probability (*prob*) to be achieved, for a given value of r, the above relationship helps determine how many models p have to be built.

From the practical perspective, we may request that the value of r should not be too low. If so, each rule-based model cannot capture the input-output dependencies. On the other hand, the excessively high dimensionality r may lead to the deterioration of the rules because of the concentration effect (and this has a detrimental impact on the clustering results). Therefore, a model built for a smaller (such as 2-3) number of input variables translate into the corresponding value of r. The plot of probability that all the variables have been selected displayed as a function of p for the selected values of r is shown in Fig. 5.2. This relationship helps determine the number of models once the value of r has been specified and the required minimal probability p has been fixed.

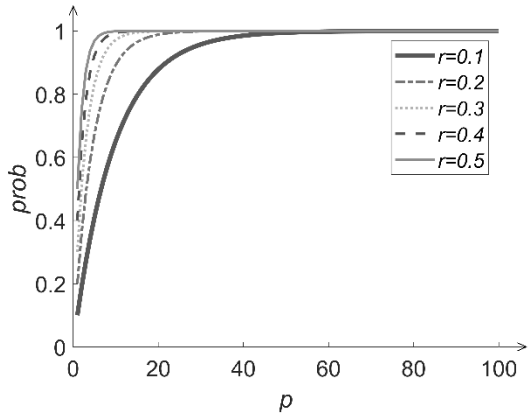


Fig. 5.2. Probability *prob* versus p for selected values of r.

From Fig. 5.2, we see that when 10% of the original features is selected each time, it becomes necessary to construct 90 models with randomly selected feature to use all of them. If the probability *prob* has been set up as 0.7, 12 models are required with $r = 0.1$ while 2 models in the case of $r = 0.5$.

Gradient boosting of the rule-based model

Each rule-based model is further refined by applying gradient boosting. The objective here is to improve the performance of the initially constructed models. Here we follow a well-known scheme of updates for the output of the model, guided by the error values [102].

Consider the j^{th} model M_j is constructed on the basis of $\boldsymbol{D}_{train,j}$. One determines the corresponding errors e_k, $k = 1,2,\dots,card(\boldsymbol{D}_{train,j})$ produced by this model and constructs an auxiliary model $M_j^{\sim}$ on the basis of input-output pairs in the format $(\boldsymbol{x}_k, e_k)$, and then aggregates the result of the model and the auxiliary construct in the additive form $M_j(\boldsymbol{x}_k) + \lambda M_j^{\sim}(\boldsymbol{x}_k)$ such that the sum of errors between the data and the aggregate above is minimized by choosing a suitable value of λ. The above boosting process is repeated K times by forming successive refinements of the augmented models to obtain the optimized model result $M_{opt,j}$.

Aggregation of partial results

The results produced for the already gradient boosted p models $M_{opt,1}, M_{opt,2}, \dots, M_{opt,p}$ are aggregated by taking a weighted average in the form

$$\hat{y} = w_1 M_{opt,1}(\boldsymbol{x}) + w_2 M_{opt,2}(\boldsymbol{x}) + \cdots + w_p M_{opt,p}(\boldsymbol{x}) \tag{5.3}$$

where $\boldsymbol{w} = [w_1, w_2, \dots, w_p]^T$ is a vector of adjustable weights of the aggregation process; the process is subject to optimization. In this optimization, the performance index is taken as a sum of squared errors. Taken from all data $\boldsymbol{D}_{train}$. As the above optimization problem concerns a standard objective function, there is an analytical solution to the optimal weights $\boldsymbol{w}_{opt}$. The objective is to minimize the distance (sum of squared errors) between the training target $\boldsymbol{target}$ and the output of the model. With the aid of the LSE minimization algorithm, the optimal weight is:

$$\boldsymbol{w} = (M^T M)^{-1} M^T \boldsymbol{target} \tag{5.4}$$

where M is an $N \times p$ matrix

$$M = \begin{bmatrix} M_{opt,1}(\boldsymbol{x}_1) & M_{opt,2}(\boldsymbol{x}_1) & \dots & M_{opt,p}(\boldsymbol{x}_1) \\ \vdots & \vdots & \ddots & \vdots \\ M_{opt,1}(\boldsymbol{x}_N) & M_{opt,2}(\boldsymbol{x}_N) & \dots & M_{opt,p}(\boldsymbol{x}_N) \end{bmatrix} \tag{5.5}$$

Then with the optimal parameter w_i, the performance index Q evaluates the quality of the aggregation of rule-based models, see (5.3).

5.3 Experimental studies

In this section, we report on the results obtained for the rule-based model designed as discussed in the previous sections. The performance of the model is reported in terms of its *RMSE* value. The data are linearly normalized to [0,1]. In the slew of experiments, we set up the following values of the parameters:

FCM: $m = 2$, the number of iterations is 100. The number of clusters c was varied from 2 to 10. We optimized the performance of the overall model by choosing the optimal number of clusters for each model. We also tried more values of c positioned in the range 2-20; no visible improvement has been reached for the values over 10. Subsequently, the range 2-10 has been selected.

Randomization: the values of r were selected as 0.1, 0.2, 0.3, 0.4, and 0.5. With the probability, $prob = 0.999$, the results coming from the theoretical analysis are $p = 84, 40, 25, 18, and\ 13$ models, respectively. The random selection of data was governed by a

uniform probability distribution. In the experiment, in order to ensure the data integrity, we set $p = 100$ for all percentage values of r.

The experiments were completed for the UCI machine learning dataset Super conductivity (https://archive.ics.uci.edu/ml/datasets/superconductivty+data), consisting of 21,263 80-dimensional data.

Following the overall design process, we randomly select some data and use them to train the individual fuzzy rule-based model. Then, the models are refined with the use of gradient boosting, see Fig. 5.3. These plots are reported with the selected percentage r being 0.5 and the number of models p being 100. It is apparent that for successive values of K, the performance index V decreases. However, the decline is reported some initial values of K, say 5-10 and then the values of the index stabilize. It is noticeable that the averaging of the outputs of the models lead to some improvement. The optimized weighted aggregation leads to better performance, as visibly displayed in Fig. 5.3.

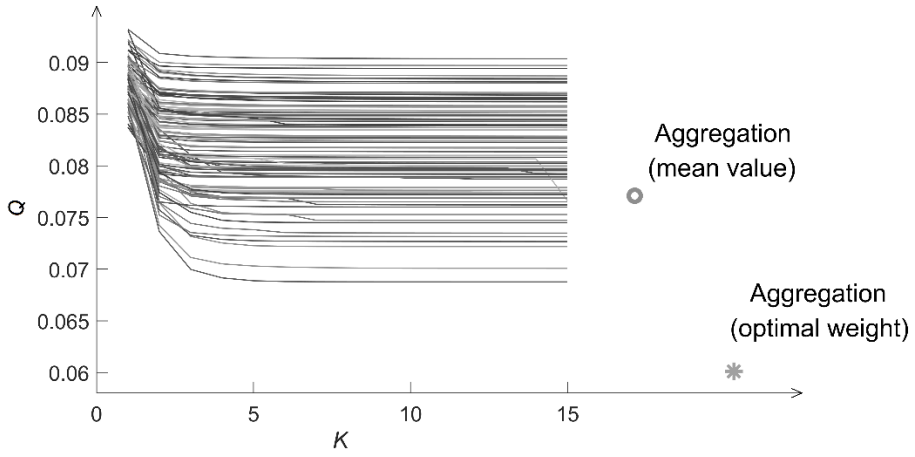


Fig. 5.3. Performance index Q obtained for each distributed model and the aggregation results ($r = 0.5$).

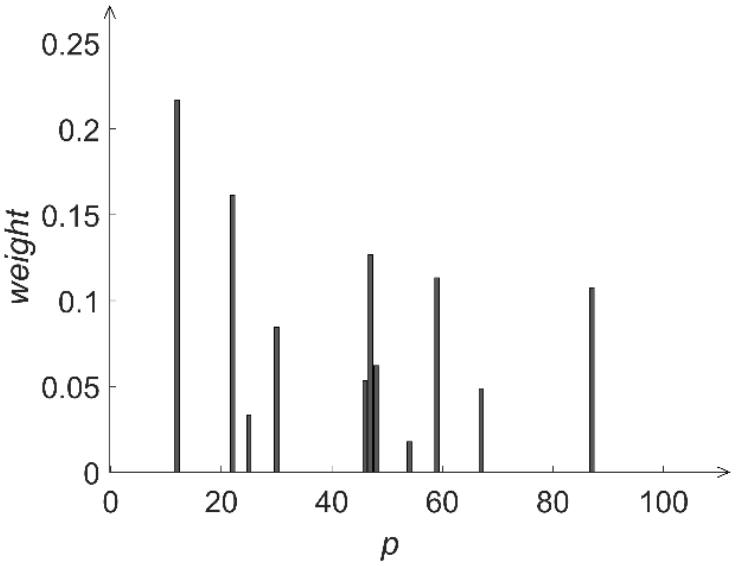


Fig. 5.4. Optimal values of the weights used in the weighted aggregation of the models.

By inspecting Fig. 5.4, we conclude that there is only a handful of models that contribute to the aggregation process while most of them exhibit a very limited impact as the values of the corresponding weights are close to zero. In Fig. 5.5(a), we show the values of the performance index (*RMSE*) for several selected values of r.

For comparative analysis, as a reference model, we consider a standard TS model. As before, *FCM* was run for 100 iterations and m was set to be 2; the results are shown in Fig. 5.5(b). The distributed rule-based model led to the improvement over the TS model; on average (across all experiments) the improvement was around 12.8%. In Fig. 5.5(c), we depict the prototypes with black circles for the TS model when c is 2 and 10. The prototypes are in the range $[0, 1]$ since the data are normalized. From the figure we see that the prototypes are close to each other. This shows that the TS model cannot effectively cluster the data, thus affecting the performance of the model.

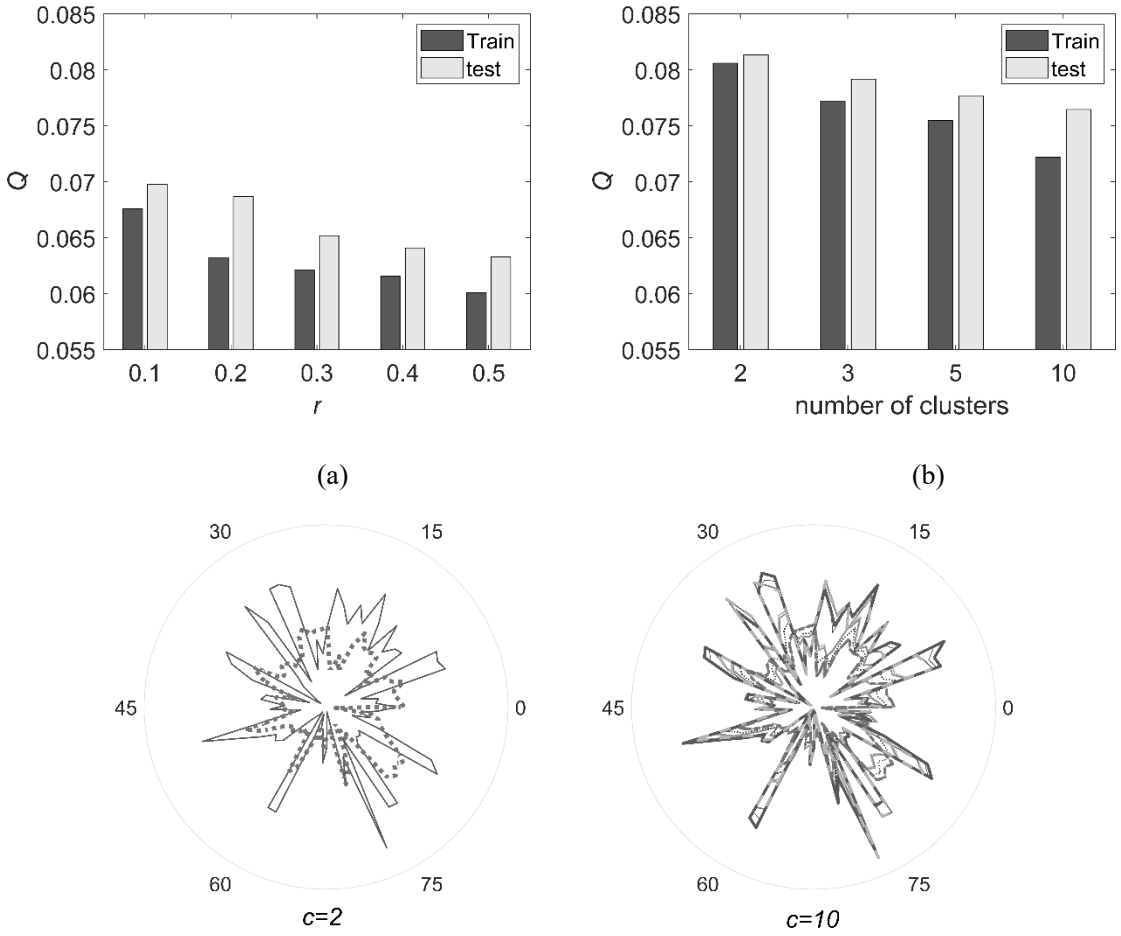


Fig. 5.5. Experimental results obtained for Super conductivity data set. (a) performance index - distributed model; (b) performance index - TS model; (c) TS model -prototypes.

5.4 Further experimental results with selected machine learning data

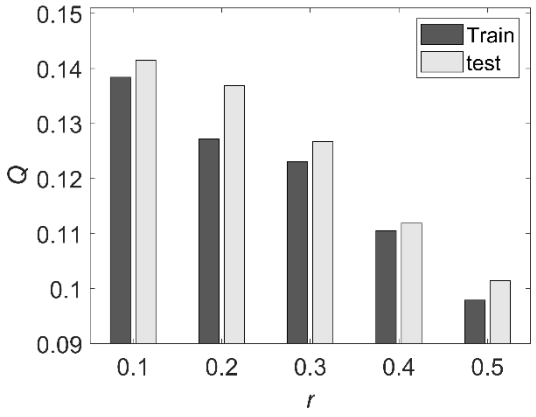
In this section, we report on experimental results obtained for some machine learning datasets coming from UCI machine learning datasets (https://archive.ics.uci.edu/ml/index.php) and Kaggle (https://www.kag-gle.com/), especially the high dimensional data: Online news popularity, Year prediction MSD and Geographical original of music.

Our intent is to show the impact of the main parameters on the performance of the obtained models as well as contrast their performance vis-à-vis a TS model constructed for all data. The details of the data are covered in Table 5.1. It is worth noting that we selected the data of the largest dimensionality of the input space as those are quite challenging in the design of rule-based models. For each data set, in order to facilitate comparison, we focus on showing the results for the gradient boosting distributed rule-based model and the reference TS model. We show the prototypes, where c is equal to 2 and 10.

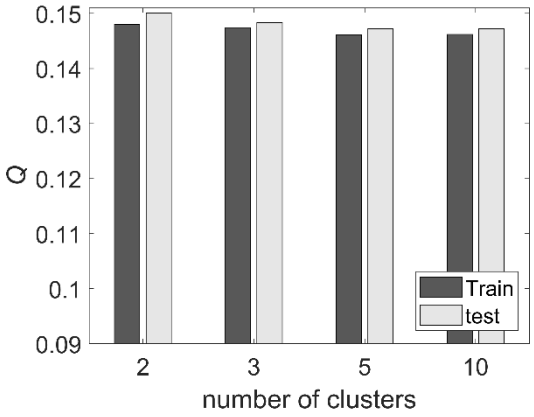
Table 5.1. Data sets used in experiments.

Data	(number of data, dimensionality of input space)
Online news popularity (https://archive.ics.uci.edu/ml/datasets/online+news+popularity)	(39,644; 58)
Year prediction MSD (first 30K data points) (https://archive.ics.uci.edu/ml/datasets/yearpredictionmsd)	(30,000; 90)
Parkinson's telemonitoring (https://archive.ics.uci.edu/ml/datasets/parkinsons+telemonitoring)	(5,875; 17)
Geographical original of music (http://archive.ics.uci.edu/ml/datasets/geographical+original+of+music)	(1,059; 117)

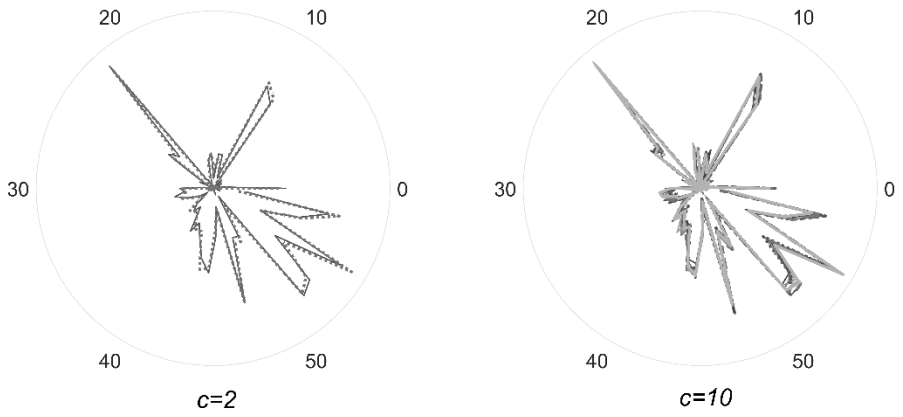
The results are displayed in a series of plots shown in Fig. 5.6.



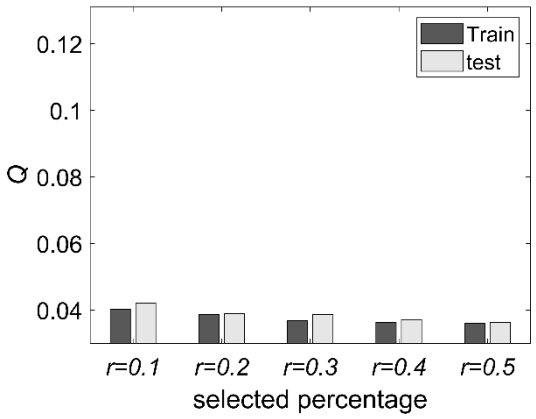
(a) i.



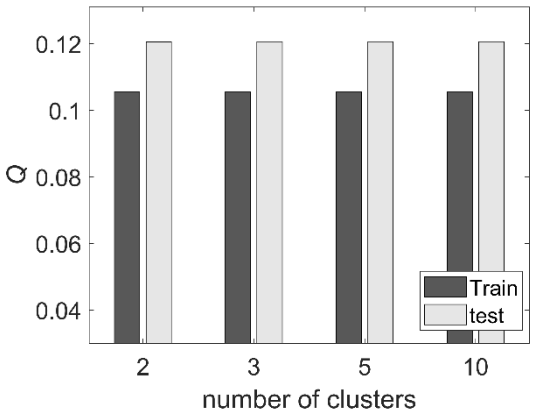
(a) ii.



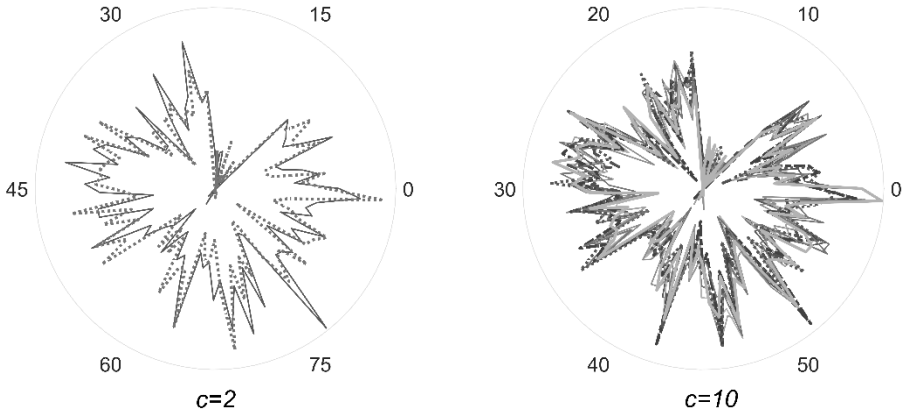
(a) iii.



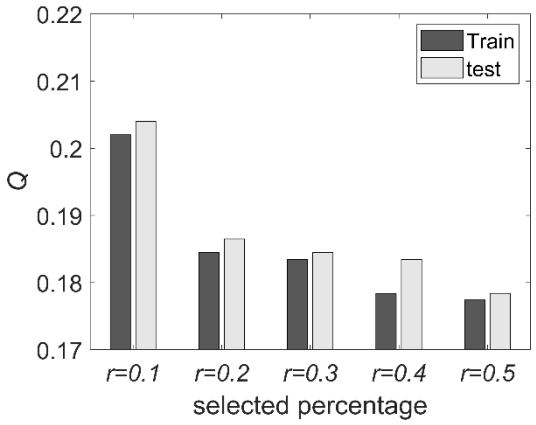
(b) i.



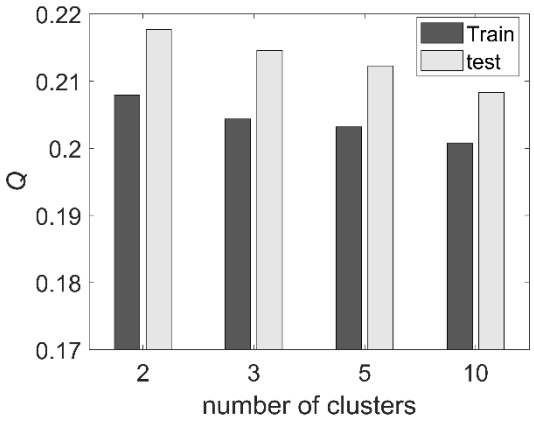
(b) ii.



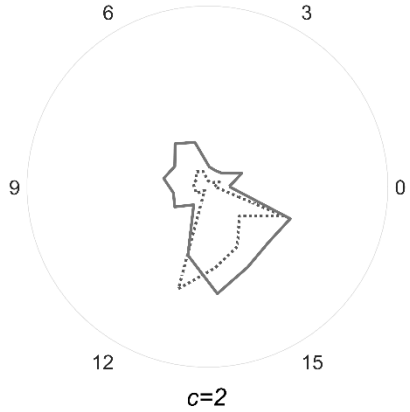
(b) iii.

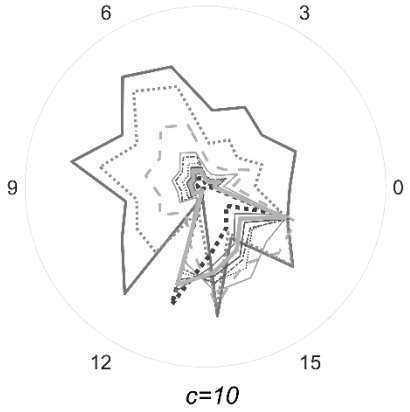


(c) i.

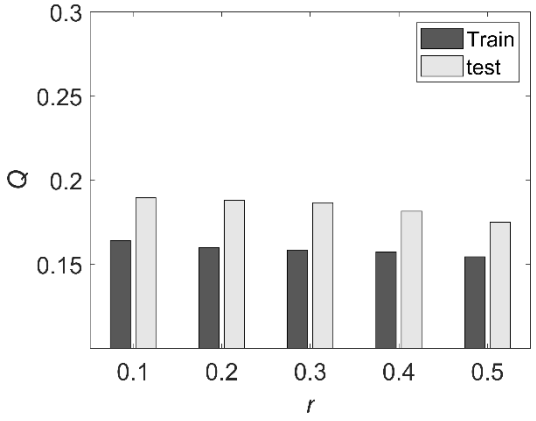


(c) ii.

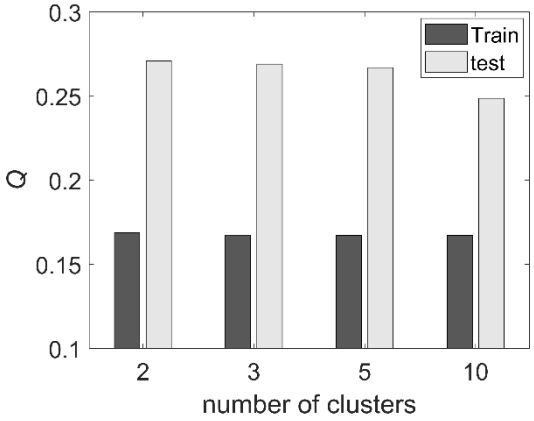




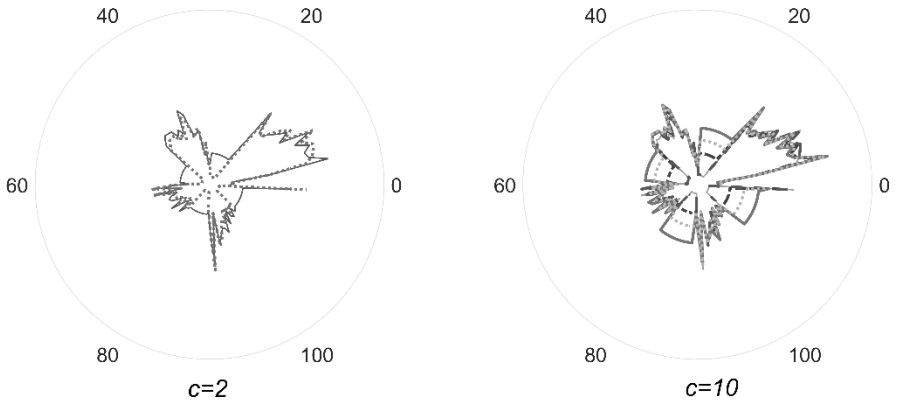
(c) iii.



(d) i.



(d) ii.



(d) iii.

Fig. 5.6. The results for different datasets. (a) Online news popularity; (b) Year prediction MSD; (c) Parkinson's telemonitoring; (d) Geographical original of music. The plots from left to right display: i. performance index - distributed model; ii. performance index - TS model; iii. radar plots present prototypes produced by the TS model.

Table 5.2 summarizes improvements of the proposed model obtained for each dataset and we compared it with the TS model with the number of clusters set to 10 (for this number of rules, the TS model produced the best results).

Table 5.2. Improvement obtained for each dataset.

Data	Improvement (%)					
	Train			Test		
	min	max	average	min	max	average
Super conductivity	6.4	16.8	12.9	8.8	17.3	13.4
Online news popularity	5.3	33.0	18.3	3.9	31.1	16.0
Year prediction MSD	61.9	65.8	64.4	65.2	69.9	68.0
Parkinson's telemonitoring	1.6	11.6	8.2	2.7	14.4	10.2
Geographical Original of Music	1.8	7.6	4.9	23.8	27.3	25.4
Average improvement (across all data)	15.4	27.0	21.7	20.9	32.0	26.6

5.5 Conclusions

In this study, we have introduced a distributed model which is developed by integrating the ideas of ensemble learning and the gradient boosting algorithm. In this process, we demonstrate how to ensure integrity of the data by randomly sampling and repeating the experiment. The distributed model obtained through the sampling process and the fuzzy rule-based model is further improved by the gradient boosting algorithm, and the improved results are aggregated. In this way, we avoid the problem of the curse of dimensionality in the face of large-dimensional data. Compared with the traditional TS model, the performance of our model has been significantly improved. In terms of accuracy, for different datasets, our optimization has reached a maximum of 31% for the gradient boosting TS model.

Chapter 6 Enhancements of Rule-Based Models through Refinements of Fuzzy C-Means

In the previous chapters, we have shown three ways to deal with high-dimensional data with fuzzy rule-based models. In this chapter, we analyze the principle of fuzzy clustering, intervene in the clustering process by adding constraints and other means, and propose a variety of effective optimization methods to improve the clustering process.

6.1 Fuzzy rule-based models and fuzzy clustering: some design highlights

In the classical rule-based model, it builds on the basis of fuzzy clustering, in particular FCM, exhibit some advantages that are associated with their modularity and interpretability since in traditional clustering algorithms, directions are not considered. However, in fuzzy modeling we are concerned about the direction. When running the clustering algorithm, we must consider distinctions between input and output variables. The underlying structure of the rules in their generic version was expressed by (2.8). These rules form a so-called 0-order Takagi-Sugeno model. The fuzzy sets forming the condition part of the rules are defined in the *n*-dimensional input space. It is apparent that the clustering algorithm determines the structure of the rule-based model. The number of rules (c) is equal to that of clusters. The fuzzification coefficient impacts the shape of the membership functions and in the sequel implies the levels of activation of the rules and the way in which the rules interact among themselves when producing the output of the model.

Typically, the FCM algorithm is realized in the $(n+1)$-dimensional space (yielding a concatenation of the input variables and the output one). This produces $(n+1)$-dimensional prototypes in the following format $[\boldsymbol{v}_i\ \boldsymbol{w}_i]^T\ \ i = 1,2,\ldots,c$. It is clear that both the fixed

conclusions and fuzzy sets contribute to the quality of the model and any possible improvements through the refinements of these two functional components of the model.

6.2 The accommodation of extreme values in the output space

As noted, the FCM clustering (as any other clustering mechanism) is direction-free (relational), implying that in carrying out the optimization of the prototypes and the partition matrix we do not distinguish between input and output variables (even though they play a different role in any modeling pursuits). In virtue of the current form of the objective function minimized during the FCM optimization process, the prototypes realize a sort of average of the data located in the input and output spaces. By inspecting (2.4), one notes that the output of the model cannot produce the results beyond the range $[w_{min}, w_{max}]$ implied by the minimal and maximal values of the prototypes $w_{min} = \min w_i$, $w_{max} = \max w_i$, which, in light of the previous observation, is included in the interval $[target_{min}, target_{max}]$, where $target_{min}$ and $target_{max}$ are respectively the extreme values assumed by the output variable that is $target_{min} = argMin_{k=1,2,\ldots,N} target_k$, $target_{max} = argMax_{k=1,2,\ldots,N} target_k$. This leads to a lack of approximation capabilities of the model in some range of output values and yields higher values of error generated by the model.

To alleviate the problem, one has to augment the distribution of the prototypes in the clustering process by (i) distributing prototypes located in the output space across the entire output space, and (ii) incorporating this knowledge hints in the clustering algorithm. Some ideas along this line were presented with the use of so-called viewpoints, see [140]. We consider the extreme values of the output space as the entries of all prototypes constructed during the optimization process. In particular, the prototypes are given in the form

$$\begin{bmatrix} \boldsymbol{v}_1 \\ w_1 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_2 \\ w_2 \end{bmatrix} \ldots \begin{bmatrix} \boldsymbol{v}_{c-2} \\ w_{c-2} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{c-1} \\ target_{min} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_c \\ target_{max} \end{bmatrix} \tag{6.1}$$

These two extreme entries ($target_{min}$ and $target_{max}$) are not optimized (updated) during the FCM optimization. They are kept intact. As a consequence, the obtained prototypes involve the extreme values as the bounds $w_{min} = target_{min}$ and $w_{max} = target_{max}$ are retained.

6.3 Reduction of spurious activation levels of rules

The membership functions produced by the FCM algorithm satisfy the obvious requirements (implied by the partition requirement) stating that for any input $\boldsymbol{x}$ a sum of membership grades is equal to 1, i.e., $\sum_{i=1}^{c} A_i(\boldsymbol{x}) = 1$. This is one of the fundamental requirements imposed on the partition matrix. While in the description of the data through a collection of clusters, this requirement does not exhibit any essential implications as to the nature of the revealed clusters. One encounters a far-reaching implication in the context of rules. The reason is the following. The above requirement states that the membership functions of A_i obtained through clustering are not unimodal but exhibit some rippling effect. The intensity of these ripples is dependent upon the values of the fuzzification coefficient used in the clustering method. Higher values of *m* intensify the obtained ripples. As the degrees of membership are in effect the activation levels of the rules and in turn produce the partial output $A_i(\boldsymbol{x})w_i$ (for rules in the form given by (2.8)), for some $\boldsymbol{x}$ being positioned quite remotely from the prototype of the i^{th} rule $\boldsymbol{v}_i$, there still might be a significant impact of this rule on the overall output.

To diminish or substantially eliminate the impact of the membership grades of the prototypes (fuzzy sets) positioned quite far from a certain input for which the rule-based mapping is realized,

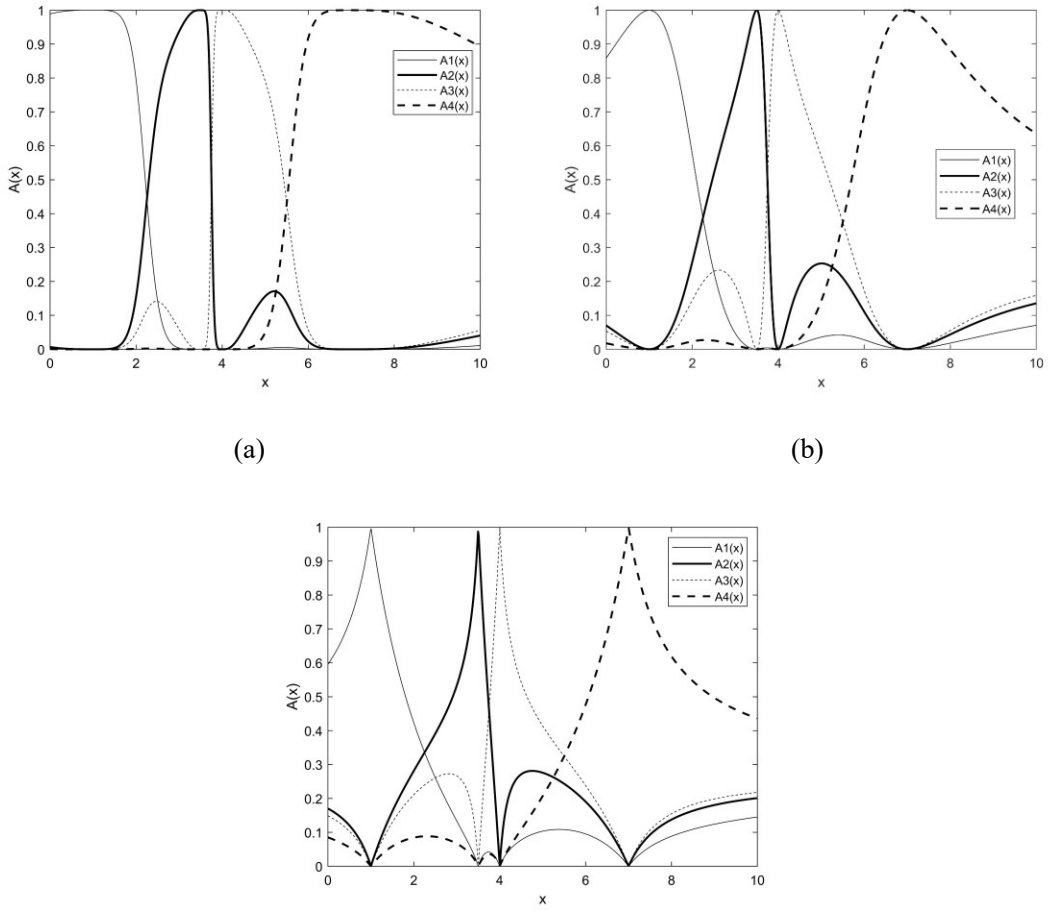
we introduce a thresholding mechanism through which lower membership grades are "filtered" out and made equal to zero. This results in a so-called λ-level fuzzy set A^{λ} constructed on a basis of A and expressed as:

$$A^{\lambda}(\boldsymbol{x}) = \begin{cases} A(\boldsymbol{x}) & \text{if } A(\mathbf{x}) \geq \lambda \\ 0, & \text{otherwise} \end{cases} \tag{6.2}$$

Using A^{λ} instead of the original membership functions, we reduce the impact of the rules (and conclusions) whose activation levels are below some adjustable (optimized) threshold level λ, i.e.,

$$\hat{y} = \frac{\sum_{\substack{i=1 \\ A_i \geq \lambda}}^{c} A_i(x)\, w_i}{\sum_{\substack{i=1 \\ A_i \geq \lambda}}^{c} A_i(\boldsymbol{x})} \tag{6.3}$$

To illustrate the thresholding effect and its impact on the computing, let us consider one-dimensional membership functions A_1, A_2, A_3, and A_4 defined over $X = [0, 10]$ and described by the prototypes $v_1 = 1$, $v_2 = 3.5$, $v_3 = 4$, and $v_4 = 7$. Using the formula for membership functions resulting from the FCM algorithm (with the values of the fuzzification coefficient *m* equal to 1.5, 2, and 3, respectively), the corresponding plots are displayed in Fig. 6.1.



(a) (b)

(c)

Fig. 6.1. Membership functions $A_1, A_2, \ldots, A_4$ obtained for different m. (a)1.5, (b) 2.0 and (c) 3.0.

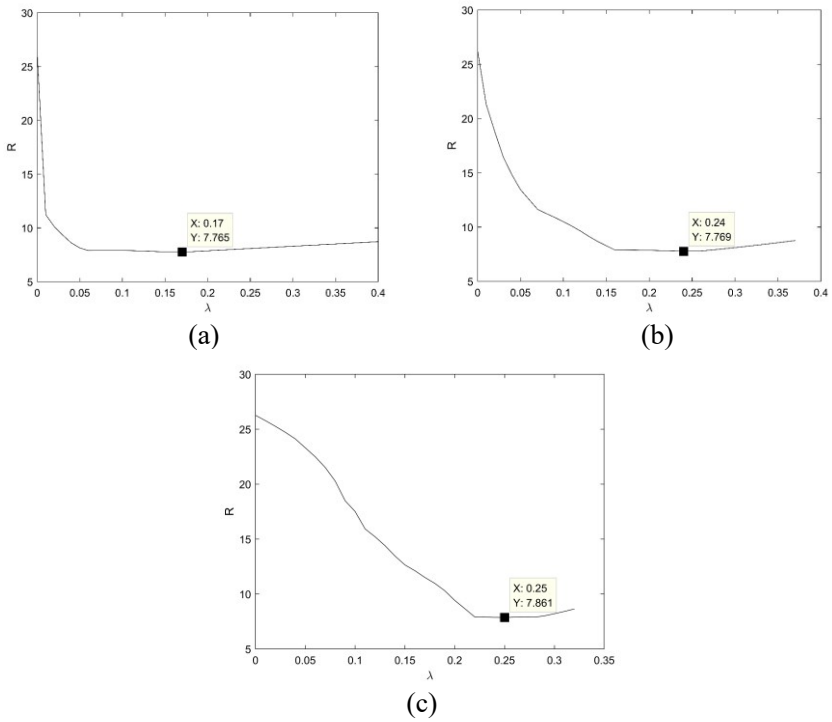
By inspecting the plots shown in Fig. 6.1, it is evident that the ripples become visible, especially for higher values of m. Now we complete a granulation-degranulation process (commonly referred to as fuzzification and defuzzification) realized as follows. We take any value of x coming from X, determine their corresponding membership grades $A_1(x)$, $A_2(x)$, $A_3(x)$, and $A_4(x)$ and then reconstruct it using the formula

$$\hat{z}(x) = \frac{\sum_{i=1}^{4} A_i^m(x) v_i}{\sum_{i=1}^{4} A_i^m(x)} \tag{6.4}$$

The value of m is the same as being used in the formation of the membership functions. When using the thresholding mechanism, the reconstructed output is described as follows where the result directly depends on the threshold:

$$\hat{z}(x,\lambda) = \frac{\sum_{i=1}^{4}(A_i^{\lambda}(x))^m v_i}{\sum_{i=1}^{4}(A_i^{\lambda})^m(x)} \tag{6.5}$$

The integral $R = \int_0^{10}|z - \hat{z}(x,\lambda)|$ serves as the reconstruction error, where the original output is $z(x) = x$. Given the adjustable value of the threshold λ, one regards R as a function of λ; $R(\lambda)$ and the optimal value λ_{opt} can be easily determined as $\lambda_{opt} = \arg Min_{\lambda} R$. The corresponding plot of R is included in Fig. 6.2 with the optimal values λ_{opt} equal to 0.17, 0.24 and 0.25 for m being 1.5, 2, and 3, respectively.

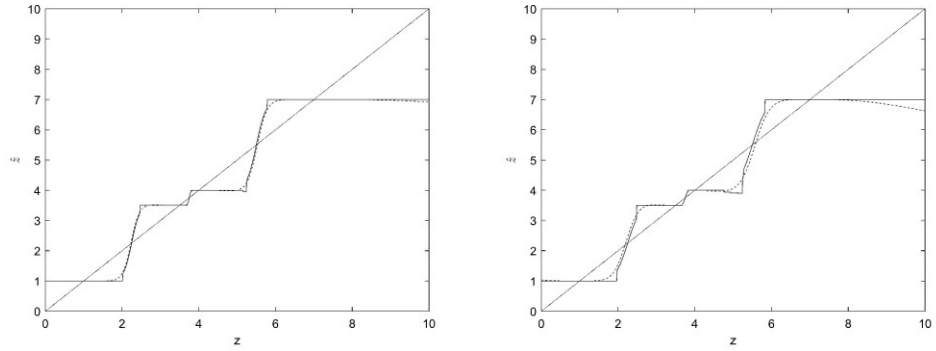


(a) (b)

(c)

Fig. 6.2. R as a function of λ for different m. (a) 1.5, (b) 2.0 and (c) 3.0.

The results of reconstruction shown in the $x - \hat{z}$ coordinates are displayed in Fig. 6.3. The optimal values of threshold obtained for $m = 1.5, 2, 3$ are $0.17, 0.24, 0.25$, respectively.



(a) (b)

(c)

Fig. 6.3. Reconstruction results in $x - \hat{z}$ coordinates.

dotted line is $\lambda = 0$ and solid line is optimal λ_{opt}. (a) $m = 1.5, \lambda_{opt} = 0.17$; (b) $m = 2.0, \lambda_{opt} = 0.24$; (c) $m = 3.0, \lambda_{opt} = 0.25$.

6.4 Quality of induced granular rules

The rules partition the input and output space into corresponding regions. The parts of regions where the membership grades are low (recall that the membership functions have infinite supports) are to be identified as not fully supported by the rules. In contrast, the regions where the membership grades are high, the regions of the spaces are fully described by the rules. To tell apart these regions from the entire space, it becomes essential to construct cores of the fuzzy rules. The

involvement of information granules in the design and analysis of rule-based models, see [70], [122], [141]–[144]. To realize this, we build set-based information granules implied by the fuzzy sets standing in the rules. Starting with numeric prototypes $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_c$ and $w_1, w_2, \ldots, w_c$, we form information granules using the principle of justifiable granularity [121], [145]–[147]. This yields interval-valued information granules $\boldsymbol{v}_i$ and w_i with radii ρ_i and τ_i, respectively. The principle of justifiable granularity is realized as follows. For each $\boldsymbol{v}_i$, we determine the product of coverage $\text{cov}(\rho_i)$ and specificity $\text{sp}(\rho_i)$ defined as follows:

$$cov(\rho_i) = \frac{1}{N} \sum_{\substack{k=1; \\ k:||x_k - v_i||^2 \le n\rho_i^2}}^{N} u_{ik} \tag{6.6}$$

where $||\boldsymbol{x}_k - \boldsymbol{v}_i||^2 = \sum_{j=1}^{n} \frac{(x_{kj} - v_{ij})^2}{\sigma_j^2}$, and

$$sp(\rho_i) = 1 - \rho_i, \rho_i \in [0,1] \tag{6.7}$$

The information granule is produced by maximizing the product of coverage and specificity with respect to radius ρ_i that is $max_{\rho_i}(cov(\rho_i)sp(\rho_i))$. The same process is realized for the prototype w_i. Thus, we obtain the corresponding information granules in the following form $V_i = (\boldsymbol{v}_i, \rho_i)$ and $W_i = (w_i, \tau_i), i = 1, 2, \ldots, c$.

The quality of the rule is assessed by looking at the specificity values of the granules V_i and W_i. The relationship between specificity values of the condition and conclusion part entails the quality of the rule. Low specificity of the condition part associated with the high specificity of the conclusion implies a high-quality rule: this rule is general (the condition embraces a lot of data

whereas the conclusion is highly detailed, viz. specific). The opposite situation, namely a high specificity of the condition part aligned with the conclusion of low specificity entails a rule of low quality. This rule applies only to a few cases and at the same time produces conclusion that is of low specificity. In light of the above observations, we express the quality of the rule q_i as follows:

$$q_i = \begin{cases} 1 & \text{if } \rho_i \geq \tau_i \\ \dfrac{\rho_i}{\tau_i} & \text{if } \rho_i < \tau_i \end{cases} \tag{6.8}$$

The higher the value of q_i, the better the quality of the rule. In this way, given the values of q_i, one can organize the rules in a linear way. Also, one can plot the values of q_i and build a histogram, which delivers an overall view at the rules. This could be done by varying the numbers of rules (the values of c are changed).

6.5 Experimental studies

Through the series of experiments, we demonstrate the impact of the refinements of the rule-based models on their performance; a special focus is on the condition part.

Let us consider one of the publicly available Machine Learning data sets, say Boston housing. (http://archive.ics.uci.edu/ml/index.php) There are several input variables and a single output variable. The dataset is randomly split into a training and testing subsets (60% for training, 40% for testing). The performance of the rule-based model is described by the Q in (2.19).

We vary the values of c from 2 to 10. The values of m varying from 1.05 to 3.0 with a step of size 0.2. The experiments are organized in the following way:

(i) Using the standard FCM, the results are collected in a tabular format showing the values of the performance index (training and testing data) for different numbers of rules; for each c, the optimal value of m, m_{opt}, is reported.

Table 6.1. Q as a function of c with optimized m.

c	2	3	4	5	6	7	8	9
m_{opt}	1.65	1.25	1.25	1.65	1.65	1.65	1.45	1.65
Q_{train}	53.81	40.30	32.15	25.71	24.72	25.81	23.87	22.85
Q_{test}	72.66	62.71	43.35	41.24	40.99	42.10	38.80	37.46

(ii) Now the experiment is repeated for the included extreme values of the output $target_{min}$ and $target_{max}$. The results are reported in the table having the same structure as the one shown above.

Table 6.2. Q as a function of c with optimized m and changed prototypes.

c	2	3	4	5	6	7	8	9
m_{opt}	2.85	2.85	2.85	2.25	2.25	2.25	2.05	2.05
Q_{train}	98.83	36.43	37.04	21.84	22.96	24.26	24.69	24.98
Q_{test}	109.55	52.30	54.03	34.47	37.13	38.90	39.53	40.71

Compare these two tables, when c is 3,5,6,7, Q with the changed prototypes is lower than that of the original FCM. We can see that this method has improved the FCM algorithm a little, and we also use this method for other datasets in the following datasets, and the results are satistied.

In Fig. 6.4, we show the change of prototypes, where the circle stands for the original prototypes, and the star means the changed prototypes.

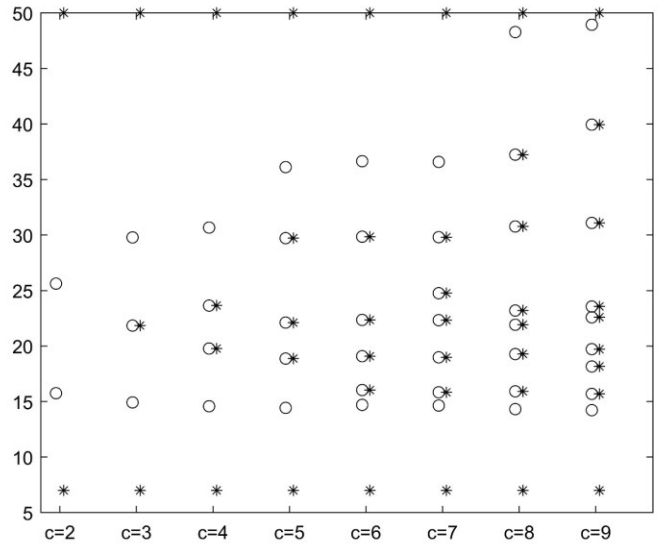


Fig. 6.4. The distribution and change of prototypes.

(iii) Here we consider the clipping effect by introducing a certain value of λ. Note that λ cannot exceed a certain maximal value such that the sum of membership functions is greater than zero. The results are reported in Table 6.3 as before (by varying the values of m and c); in the table one reports the best value of Q for the training and testing data (viz. the one obtained when varying the values of λ).

Table 6.3. Performance of original and clipping models with optimized λ.

m/c		2	3	4	5	6	7	8	9
1.05	Q_{train}	56.37	40.61	34.85	33.72	33.27	29.21	25.31	27.65
	$Q_{train}cut$	56.37	40.61	34.84	33.65	33.27	29.21	25.31	27.65
	Q_{test}	76.63	64.33	46.20	47.92	45.86	49.40	40.28	43.17
	$Q_{test}cut$	76.63	64.33	46.21	48.02	45.86	49.40	40.28	43.17
	λ	0.00	0.00	0.06	0.21	0.00	0.00	0.00	0.00
1.25	Q_{train}	55.09	40.30	32.15	31.18	29.61	28.21	25.21	26.09
	$Q_{train}cut$	55.09	40.20	32.15	31.18	29.61	28.21	25.21	26.09
	Q_{test}	74.65	62.71	43.35	43.77	46.71	46.71	41.43	40.47
	$Q_{test}cut$	74.65	62.85	43.35	43.77	46.71	46.71	41.43	40.47
	λ	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.00
1.45	Q_{train}	54.12	41.82	44.95	26.42	24.74	26.41	24.88	23.75
	$Q_{train}cut$	53.98	41.59	44.71	26.41	24.71	26.40	24.88	23.66

	Q_{test}	73.11	63.09	66.91	41.58	41.54	43.66	41.57	38.90
	$Q_{test}cut$	73.05	63.33	66.81	41.52	41.72	43.83	41.57	39.05
	λ	0.02	0.12	0.05	0.01	0.04	0.03	0.00	0.03
1.65	Q_{train}	53.81	43.47	46.48	25.71	24.72	25.81	23.87	22.85
	$Q_{train}cut$	53.24	42.88	45.60	25.25	24.15	25.16	23.44	22.25
	Q_{test}	72.66	63.94	67.70	41.24	40.99	42.10	38.80	37.46
	$Q_{test}cut$	72.16	63.65	67.26	41.31	41.89	42.99	40.44	38.30
	λ	0.08	0.12	0.09	0.08	0.13	0.10	0.09	0.09
1.85	Q_{train}	54.08	44.88	46.51	26.46	25.43	26.24	25.55	25.59
	$Q_{train}cut$	52.83	43.53	45.18	24.94	23.69	24.78	23.93	24.01
	Q_{test}	72.98	64.96	67.07	42.28	41.77	42.58	41.91	41.57
	$Q_{test}cut$	71.96	64.19	67.17	40.31	41.70	42.02	42.69	39.42
	λ	0.13	0.17	0.18	0.13	0.15	0.11	0.12	0.07
2.05	Q_{train}	54.72	46.16	47.36	28.23	27.13	27.98	27.64	28.00
	$Q_{train}cut$	52.75	43.96	45.14	24.97	23.51	24.60	24.03	24.57
	Q_{test}	73.73	66.10	67.95	44.57	43.82	44.78	44.46	45.66
	$Q_{test}cut$	71.83	65.48	69.40	40.11	42.62	40.86	43.12	42.14
	λ	0.18	0.22	0.22	0.16	0.17	0.12	0.13	0.10
2.25	Q_{train}	55.54	47.39	48.47	29.50	29.59	30.64	30.53	31.19
	$Q_{train}cut$	52.73	44.48	45.43	23.86	23.54	24.60	24.28	24.05
	Q_{test}	74.70	67.29	69.13	45.60	46.81	48.11	48.02	48.96
	$Q_{test}cut$	72.13	65.78	68.97	40.25	42.55	40.55	39.56	41.53
	λ	0.22	0.23	0.22	0.18	0.18	0.13	0.10	0.12
2.45	Q_{train}	56.46	48.59	49.69	32.46	32.35	33.58	33.46	33.45
	$Q_{train}cut$	52.77	44.70	46.17	24.02	23.67	24.64	24.40	24.50
	Q_{test}	75.78	68.51	70.41	49.01	50.17	51.73	51.62	52.46
	$Q_{test}cut$	72.28	66.31	67.69	40.44	41.62	42.32	41.26	43.03
	λ	0.25	0.24	0.16	0.19	0.18	0.14	0.13	0.11
2.65	Q_{train}	57.42	49.74	50.93	35.53	35.12	36.47	35.73	35.35
	$Q_{train}cut$	53.59	44.93	46.43	24.41	24.05	25.07	24.71	24.57
	Q_{test}	76.87	69.70	71.69	52.58	53.52	55.24	54.36	54.09
	$Q_{test}cut$	73.34	66.53	70.91	40.21	41.71	42.64	42.01	42.08
	λ	0.26	0.25	0.25	0.20	0.18	0.16	0.13	0.11
2.85	Q_{train}	58.37	50.85	52.12	38.51	37.83	39.34	34.85	33.24
	$Q_{train}cut$	54.89	45.10	46.65	24.84	24.54	25.37	23.47	22.39
	Q_{test}	77.96	70.85	72.93	56.04	56.74	58.63	52.89	51.41
	$Q_{test}cut$	73.13	67.00	71.08	43.52	42.43	43.21	39.08	39.99
	λ	0.26	0.25	0.25	0.21	0.18	0.16	0.14	0.12

From Table 6.3, we can see that the performance index of the original FCM is higher than that one encounters in the case of using an optimal value of λ; the improvements are in the range of 0- 35.5 %, and the small improvements only exist for small m. With the increase of m, Q decreases a lot. Table 6.4 reports the details on the improvements of Q regarded as a function of c and m.

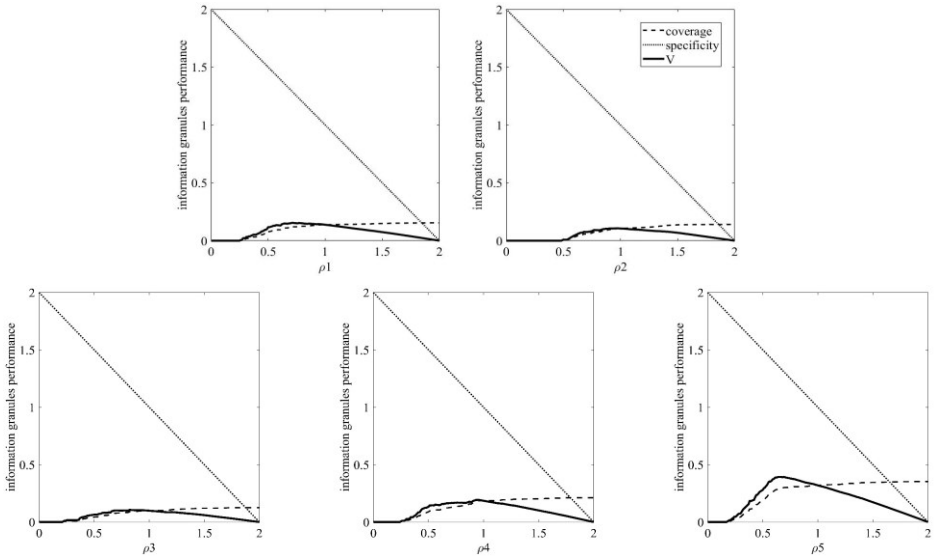
Table 6.4. Performance improvements of fuzzy model with the clipping effect.

m/c		2	3	4	5	6	7	8	9
1.05	train	0.0%	0.0%	0.0%	0.2%	0.0%	0.0%	0.0%	0.0%
	test	0.0%	0.0%	0.0%	-0.2%	0.0%	0.0%	0.0%	0.0%
1.25	train	0.0%	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	test	0.0%	-0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
1.45	train	0.3%	0.6%	0.5%	0.0%	0.1%	0.0%	0.0%	0.4%
	test	0.1%	-0.4%	0.1%	0.1%	-0.4%	-0.4%	0.0%	-0.4%
1.65	train	1.1%	1.4%	1.9%	1.8%	2.3%	2.5%	1.8%	2.6%
	test	0.7%	0.5%	0.6%	-0.2%	-2.2%	-2.1%	-4.2%	-2.2%
1.85	train	2.3%	3.0%	2.9%	5.7%	6.8%	5.6%	6.3%	6.2%
	test	1.4%	1.2%	-0.1%	4.7%	0.2%	1.3%	-1.9%	5.2%
2.05	train	3.6%	4.8%	4.7%	11.5%	13.3%	12.1%	13.1%	12.3%
	test	2.6%	0.9%	-2.1%	10.0%	2.7%	8.8%	3.0%	7.7%
2.25	train	5.1%	6.1%	6.3%	19.1%	20.4%	19.7%	20.5%	22.9%
	test	3.4%	2.2%	0.2%	11.7%	9.1%	15.7%	17.6%	15.2%
2.45	train	6.5%	8.0%	7.1%	26.0%	26.8%	26.6%	27.1%	26.8%
	test	4.6%	3.2%	3.9%	17.5%	17.0%	18.2%	20.1%	18.0%
2.65	train	6.7%	9.7%	8.8%	31.3%	31.5%	31.3%	30.8%	30.5%
	test	4.6%	4.5%	1.1%	23.5%	22.1%	22.8%	22.7%	22.2%
2.85	train	6.0%	11.3%	10.5%	35.5%	35.1%	35.5%	32.7%	32.6%
	test	6.2%	5.4%	2.5%	22.3%	25.2%	26.3%	26.1%	22.2%

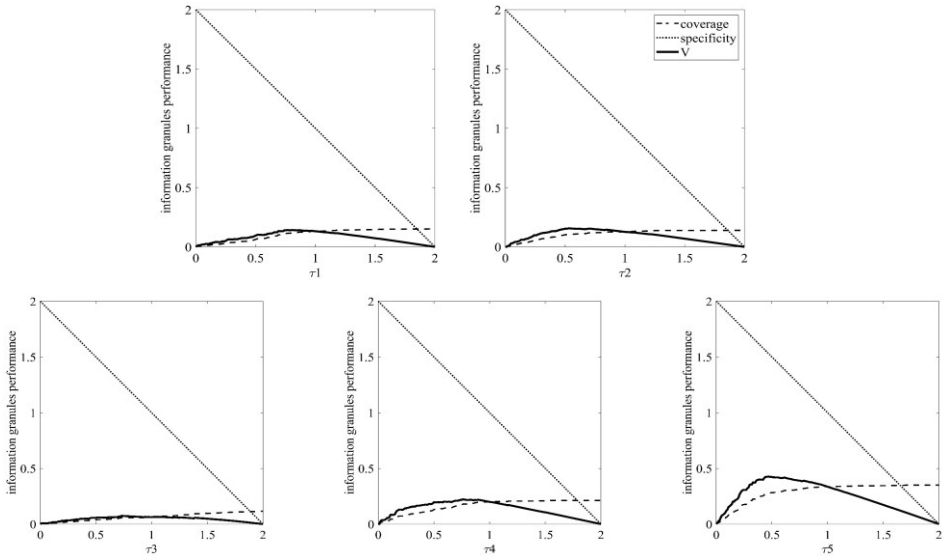
(iv) At this phase, granular rules are considered. We choose the best values of ρ and τ for $(\boldsymbol{v}_i, \rho_i)$ and (w_i, τ_i), where $i = 1, 2, 3, \ldots, c$. Here we show the result of Boston housing data (when c is 5 with the optimized $m = 2$) as an example, see Fig. 6.5. Firstly, we obtain the best values of ρ_i and τ_i by maximizing the values of V (recall that we take the product of coverage and specificity). Then we use the obtained parameters to calculate the quality of the granular model.

Figs. 6.6 and 6.7 visualize two different forms of results. Fig. 6.6 shows the contrast of ρ and τ, and Figs. 6.7 depicts the quality of the individual rules.

Fig. 6.5(a) and (b) show the performance of information granules (namely coverage, specificity and V) regarded as a function of ρ_i and τ_i.



(a)



(b)

Fig. 6.5. Information granular performance with ρ_i and τ_i. (a) ρ_i; (b) τ_i.

Fig. 6.6 depicts the relationship between ρ_i and τ_i obtained for the training data with c being set to 5. In light of (2.19), the more points are located below main diagonal, the better the quality of the model is. Fig. 6.7 displays the quality of the models for various values of c.

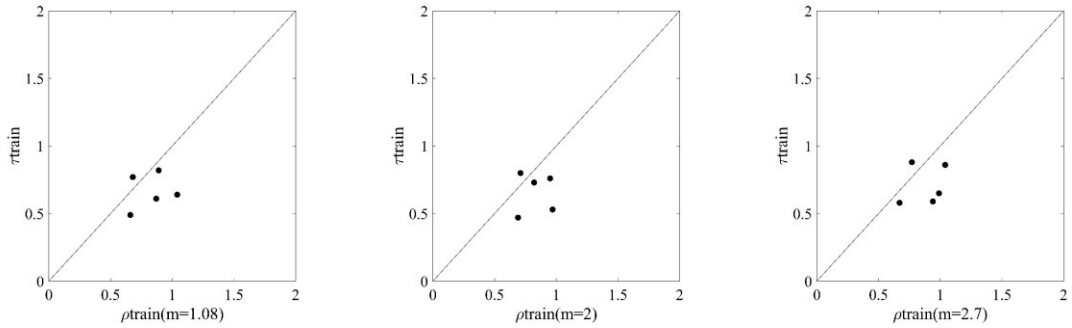


Fig. 6.6. Values of τ versus the associated values of ρ.

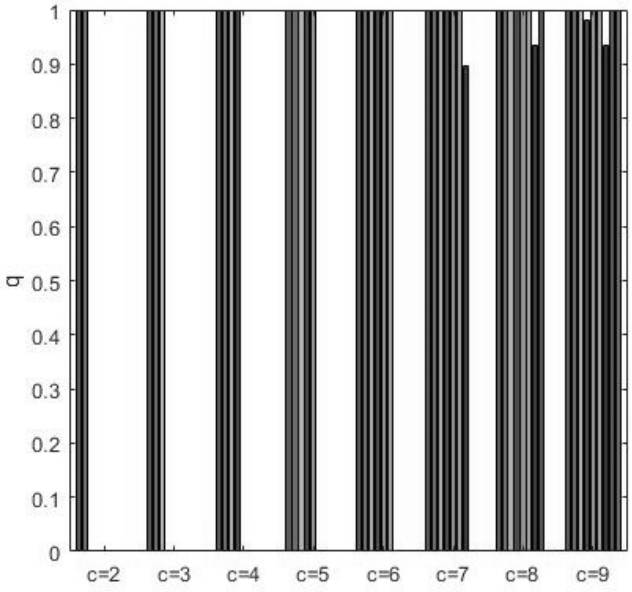


Fig. 6.7. Quality of models for selected numbers of rules.

Considering Figs. 6.6 and 6.7, when we divide Boston data into 3 clusters, the performance q is always equal to 1; that is to say, the quality of the granular rule model is very nice, and in Fig. 6.6, we have given an example of the relationship between ρ and τ when c is 5 (general case). It

is obvious that most points in the figure are below the straight line, which is another way of saying that for most rules q is equal to 1.

Then we do the same experiments and same process for other datasets in the following part. As mentioned in the introductory section, there are three improved models: (i) extreme values of the output stands for the prototypes of output with FCM, (ii) elimination of spurious activation levels of the rules, (iii) introduction of granular rules to evaluate the quality of rules. For each dataset, we present the results:

(a) Values of Q shown as a function of c with optimized m and the model with the original FCM;

(b) Q as a function of c with certain m and λ for (ii) and the original model without clipping;

(c) τ as a function of ρ for training data, each point stands for the quality criterion for each rule, and as we mentioned, more points below the straight line means that the quality is good.

(d) The quality of models for each cluster.

MAGIC Gamma Telescope Data Set

(https://archive.ics.uci.edu/ml/datasets/magic+gamma+telescope). In Fig. 6.8(b), m is 2.05 and λ is 0.06. In Fig. 6.8(c), c is 20.

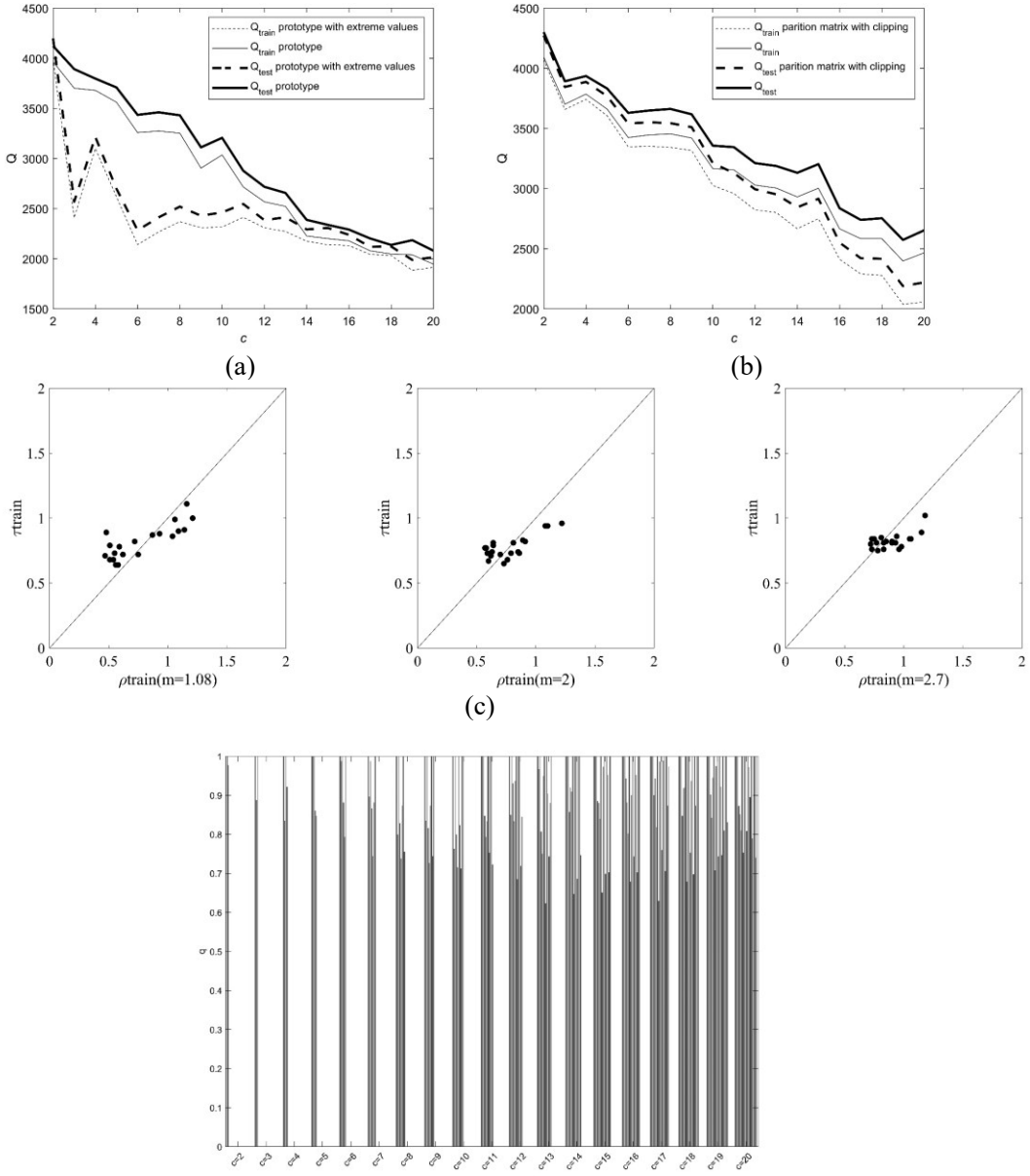


Fig. 6.8. The results for *MAGIC Gamma Telescope Data Set*. Q as a function of c with optimization of FCM: (a) Extreme values; (b) Clipping. (c) Values of τ versus the associated values of ρ. (d) Quality of models.

Wine Quality Data Set

(https://archive.ics.uci.edu/ml/datasets/wine+quality). The results are shown in Fig. 6.9 for several selected values of m.

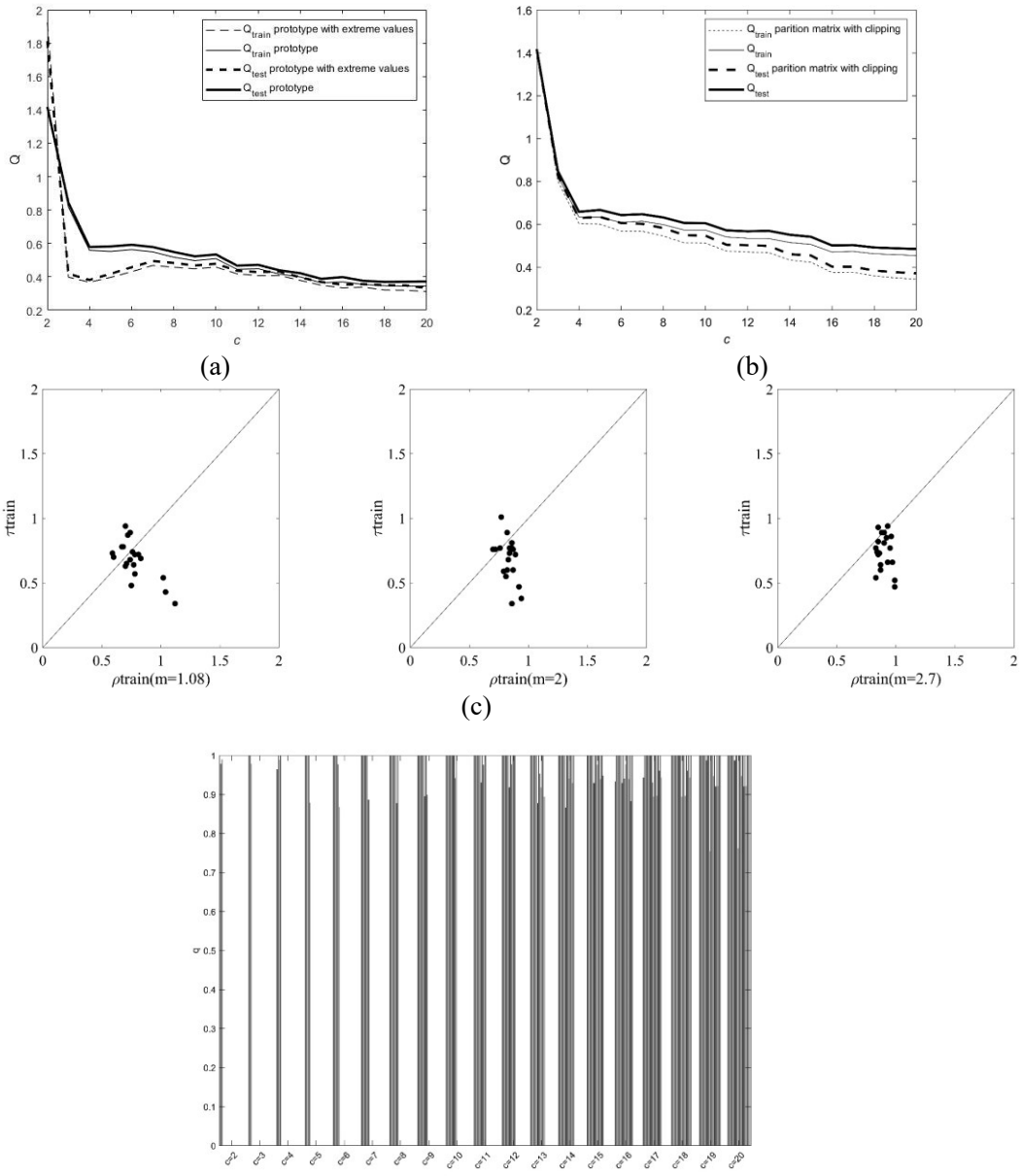


Fig. 6.9. The results for *Wine Quality Data Set*. Performance Q as a function of c with optimization of FCM: (a) Extreme values; (b) Clipping. (c). Plot of values of τ versus the associated values of ρ. (d) Quality of models for selected values of c.

Wall-Following Robot Navigation Data Data Set (https://archive.ics.uci.edu/ml/datasets/Wall-Following+Robot+Navigation+Data). The results are displayed in Fig. 6.10.

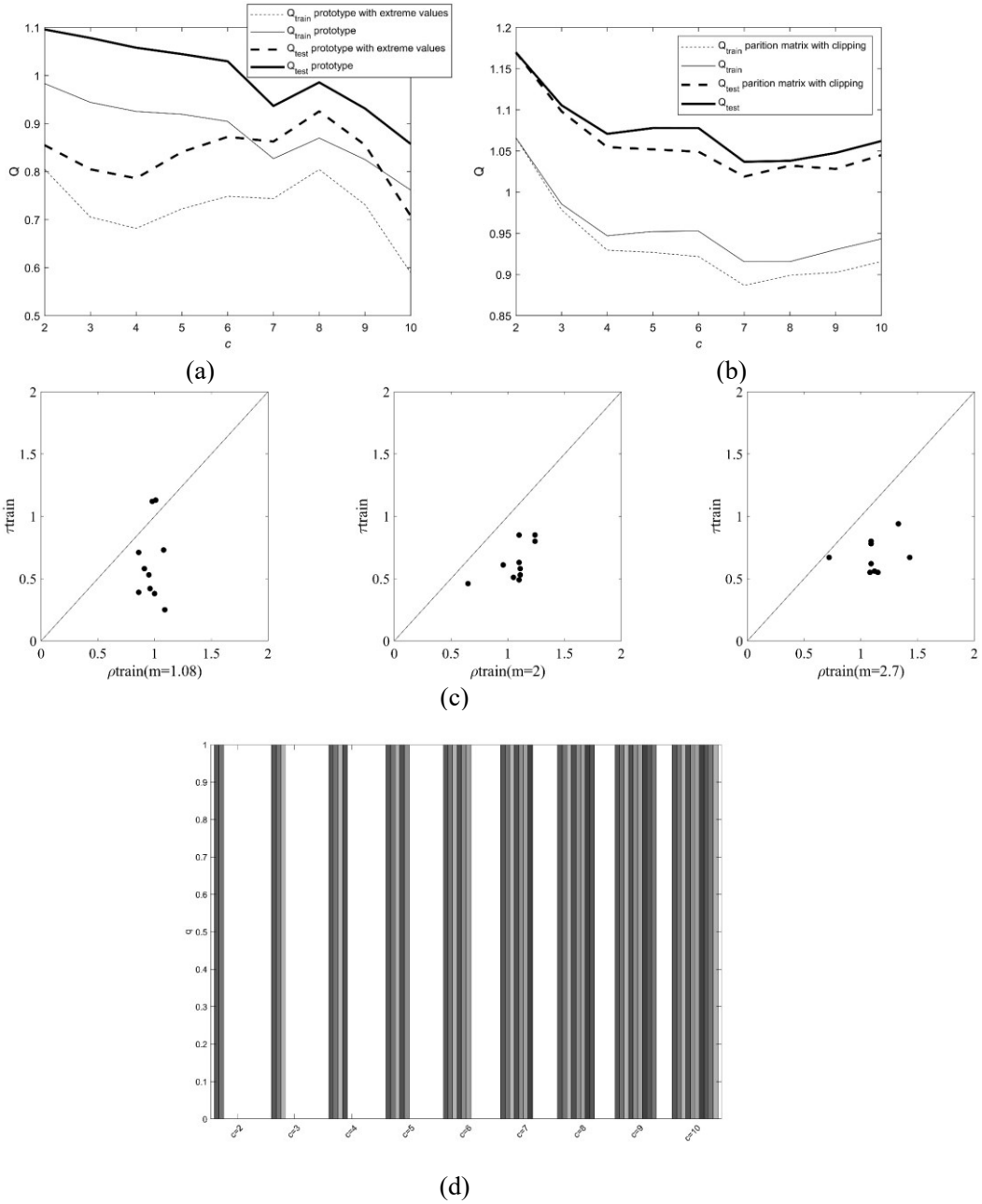


Fig. 6.10. The results for *Wall-Following Robot Navigation Data Set*. Q as a function of c with optimization of FCM: (a) Extreme values; (b) Clipping. (c) Values of τ versus the associated values of ρ. (d) Quality of models.

Banknote authentication Data Set

(https://archive.ics.uci.edu/ml/datasets/banknote+authentication). The corresponding results are displayed in Fig. 6.11.

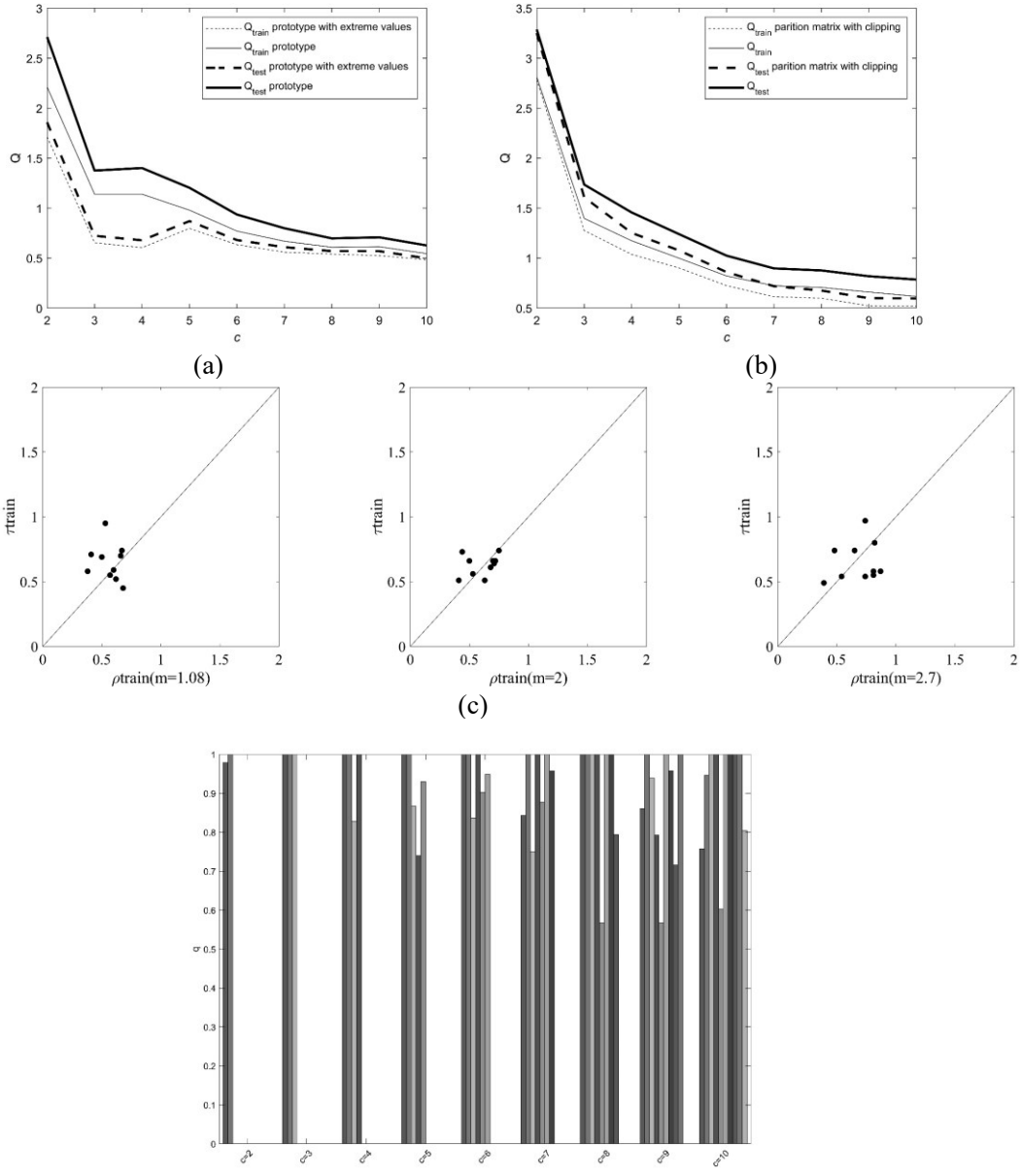


Fig. 6.11. The results for *Banknote authentication Data Set*. Q as a function of c with optimization of FCM: (a) Extreme values; (b) Clipping. (c) Values of τ versus the associated values of ρ. (d) Quality of models.

Image Segmentation Data Set

(http://archive.ics.uci.edu/ml/datasets/image+segmentation). The obtained results are reported in Fig. 6.12.

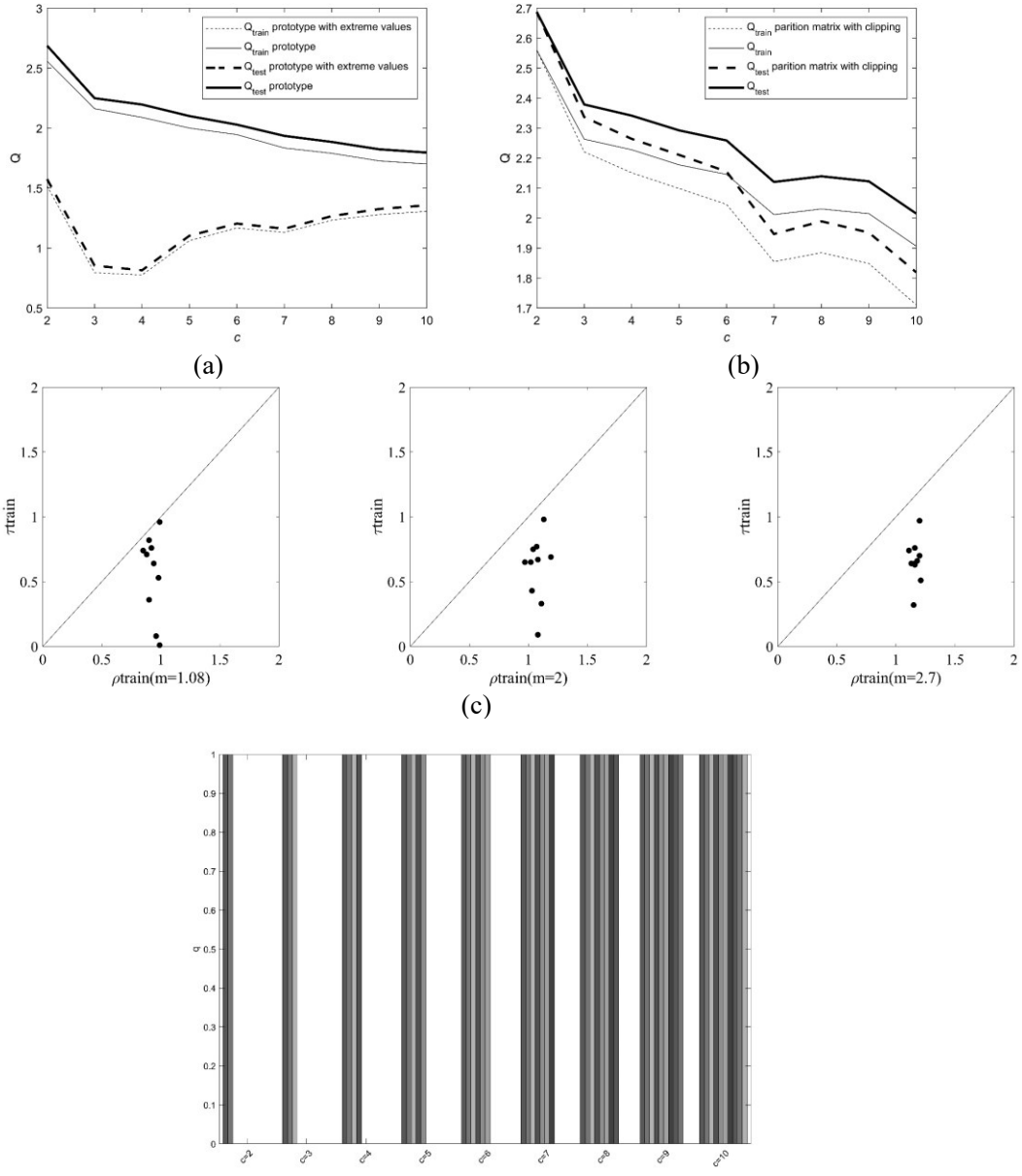


Fig. 6.12. The results for *Image Segmentation Data Set*. Q as a function of c with optimization of FCM: (a) Extreme values; (b) Clipping. (c) Values of τ versus the associated values of ρ. (d) Quality of models.

Pima Indians Diabetes Data Set

(https://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabetes/). In Fig. 6.13(b), m is 2.05 and λ is 0.14. In Fig. 6.13(c), c is 10.

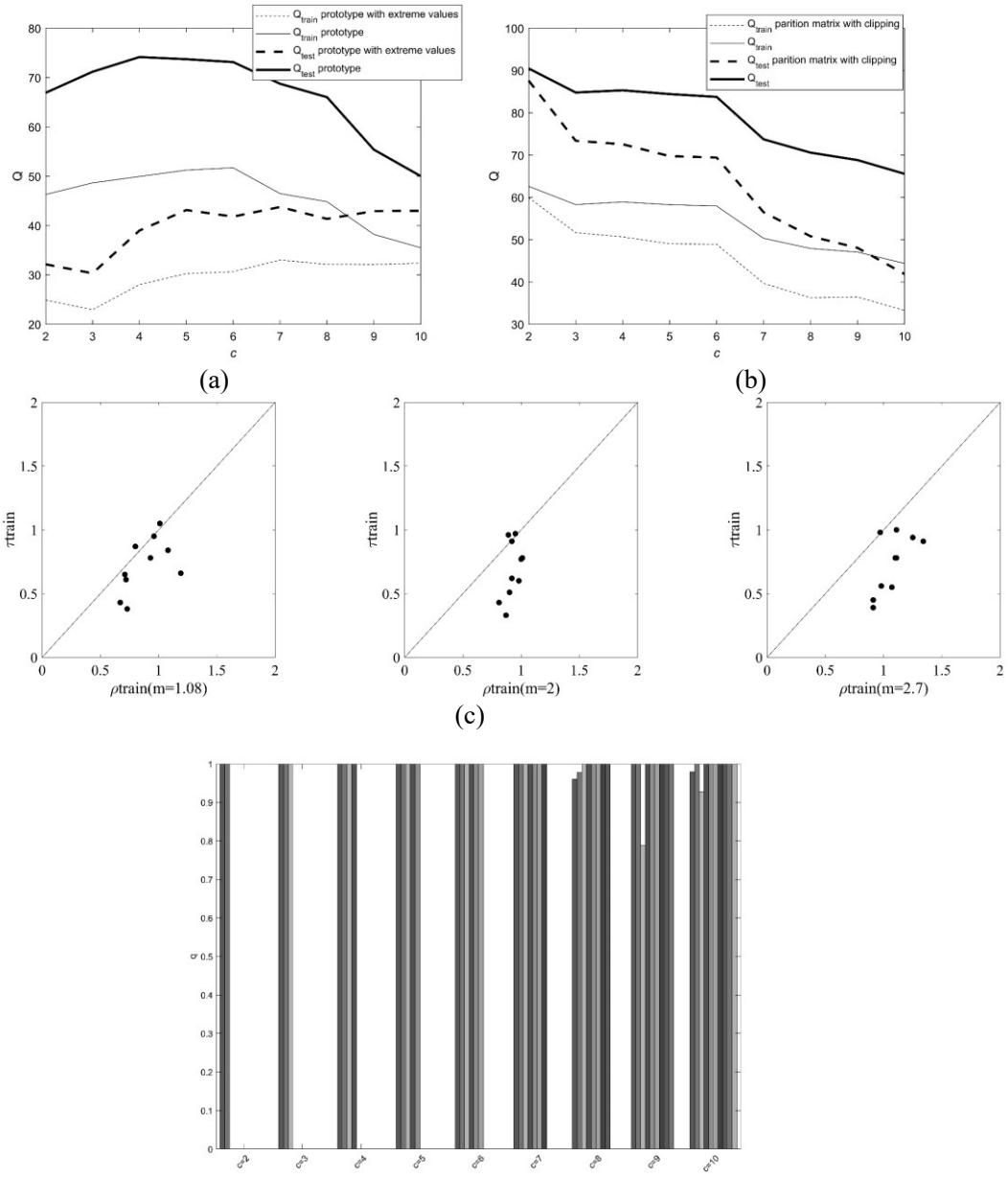


Fig. 6.13. The results for *Pima Indians Diabetes Data Set.* Q as a function of c with optimization of FCM: (a) Extreme values; (b) Clipping. (c) Plot of values of τ versus the associated values of ρ. (d) Quality of models.

Wine Data Set

(https://archive.ics.uci.edu/ml/datasets/wine). The results are reported in Fig. 6.14.

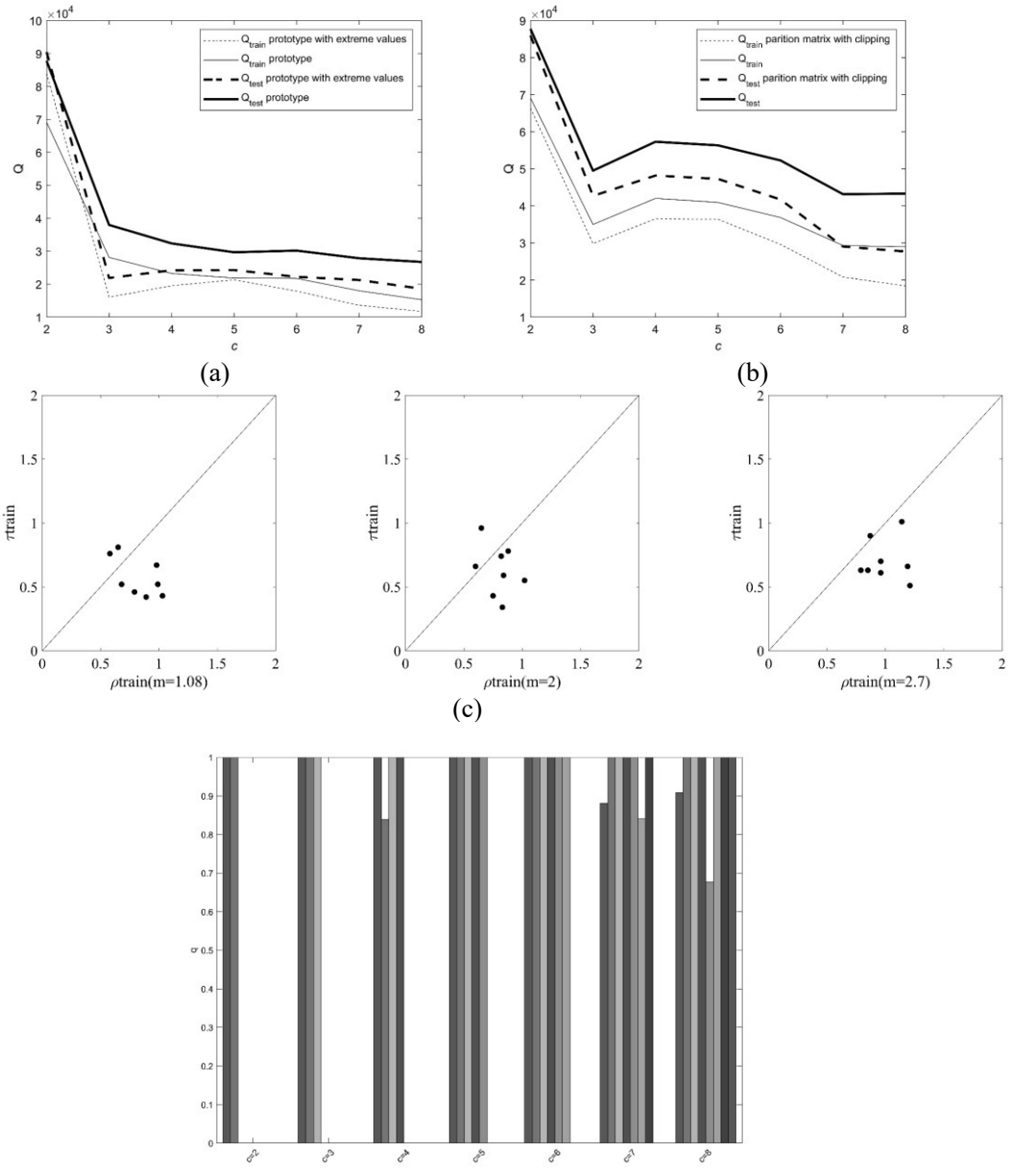


Fig. 6.14. The results for *Wine Data Set*. Q as a function of c with optimization of FCM: (a) Extreme values; (b) Clipping. (c) Values of τ versus the associated values of ρ. (d) Quality of models.

Seeds Data Set

(https://archive.ics.uci.edu/ml/datasets/seeds). The obtained results are reported in Fig. 6.15.

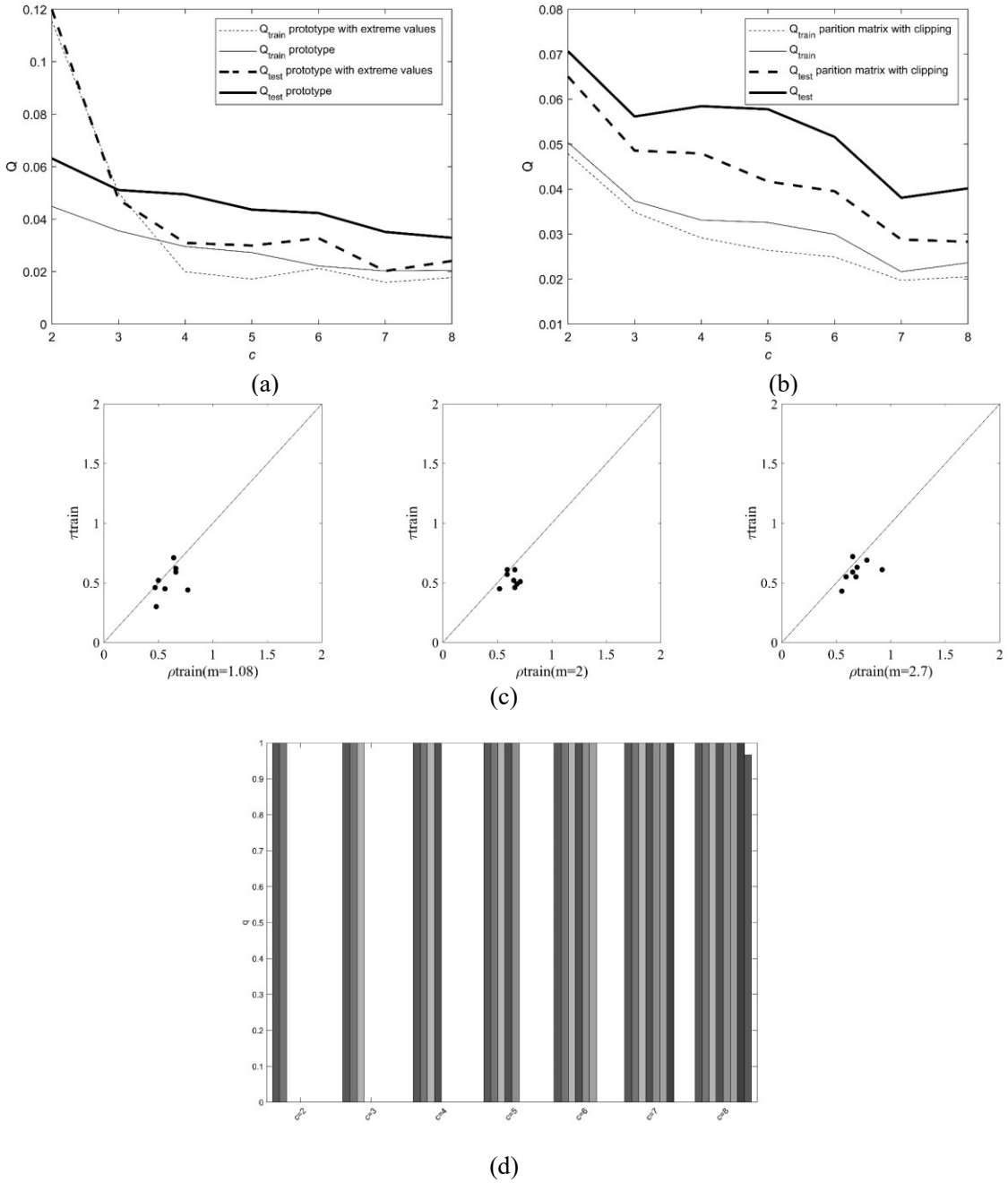


Fig. 6.15. The results for *Seeds Data Set*. Q as a function of c with optimization of FCM: (a) extreme values; (b) clipping. (c) Values of τ versus the associated values of ρ. (d) quality of models.

Based on Figs. 6.8 – 6.15, it becomes clear that the proposed refinements made to the design process give rise to the improvements of the results. The levels of improvement can vary, though and might be associated with the nature of the data. In what follows, Table 6.5, summarizes the main results where the approaches studied so far have been carefully assessed (method 1—incorporation of minimal and maximal values of the output space; method 2 - clipping spurious activation levels of the rules). The improvement level is quantified by reporting the maximal (max) and the average (mean) improvement levels.

Table 6.5. Percentage of improvement achieved for different datasets.

	Method 1		Method 2		
	max	mean	max	mean	λ range
MAGIC Gamma Telescope Data Set	35.0%	13.2%	16.6%	6.4%	0-0.14
Wine Quality Data Set	51.5%	11.2%	24.3%	12.5%	0-0.06
Wall-Following Robot Navigation Data	26.3%	17.8%	8.9%	5.1%	0-0.06
Banknote authentication Data Set	47.0%	22.4%	21.0%	12.2%	0-0.21
Image Segmentation Data Set	63.2%	41.4%	10.3%	5.2%	0-0.13
Pima Indians Diabetes Data	52.9%	34.1%	25.1%	17.1%	0-0.22
Wine Data Set	43.0%	15.1%	36.5%	18.3%	0-0.13
Seeds Data Set	37.0%	21.8%①	19.0%	11.6%	0-0.13

⋆calculations are completed for the number of clusters varying from 4 to 8.

6.6 Conclusions

Fuzzy rule-based models have been a subject of intensive design along with a variety of development strategies. In light of the central role played by fuzzy clusters in the structurization of "if-then" rules, enhancements of clustering methods can effectively improve the performance of the fuzzy rule-based model. The research reported here embarks on the pursuits cast within this general framework by offering several key augmentations of fuzzy artifacts delivered by the Fuzzy C-Means algorithm, namely an incorporation of extreme values of the output space and elimination of undesired spurious activation levels of the individual rules. The idea of granular rules, which

constitutes another interesting alternative to enhance the transparency of the rules by building a family of information granules of limited supports, sheds light on the most essential regions in the data space and identifies the most crucial relationships between input and output spaces.

Chapter 7 Conclusions and Future Works

In this chapter, we cover the main conclusions of the study and identify a number of suggestions for future research.

7.1 Conclusions

Overall, regarding the limitation of rule-based models in the high-dimensional systems mentioned in the introduction, we have proposed some improved algorithms and models and obtained a number of interesting conclusions:

(1) The matrix factorization is a commonly used method of dimensionality reduction of data. In our study, we proposed a concept of relational matrix factorization which reduces the dimensionality by engaging mechanisms of logic operations *t*-norms and *t*-conorms and improving the model interpretability. In addition, the comparative experimental studies demonstrate that our algorithm exhibits better reconstruction performance.

(2) Link fuzzy relational factorization and fuzzy rule-based models to prevent the model form dealing directly with high dimensional data. The condition part of the rules in the model is formed by the factorization process, instead of the fuzzy clustering, which improves the reliability of the fuzzy rule-based model for the high-dimensional system.

(3) Using random sampling and ideas from the random forest algorithm and gradient boosting algorithm, we developed a distributed rule-based model. Each distributed model processes the data in a small module, thus reducing the load of the model. Experiments demonstrate that our model has a certain improvement over the traditional TS model when being designed in the presence of high-dimensional data.

(4) Since the construction of rule-based models is usually accompanied by fuzzy clustering (FCM); an improved FCM can help the rule-based models to obtain better results. In this study, we proposed some augmentation methods for FCM with the use of extreme values of output space and the information granule, etc. These enhancements have been shown successful as exemplified through a series of carefully structured experiments.

In general, our research revolves around high-dimensional data and fuzzy rule-based models. The constructed model is adapted to high-dimensional systems through dimensionality reduction and the improvement of algorithms.

7.2 Future studies

Based on our current research, the following topics deserve further studies and exploration:

We propose a model based on the rule-based model in combination with fuzzy relation factorization. While the study has covered the concepts, algorithms, and delivered experimental studies along with some comparative studies (which demonstrated the usefulness of the approach), there are a number of directions worth further exploration. Parametric studies along the line of experimenting with various triangular norms and conorms could lead to additional improvements. The encoding process has been discussed, yet there is more room here to optimize families of reference information granules to enhance the performance of the constructed rules.

In our research, through the process of matrix decomposition, we effectively extract the information hidden in the data and make use of it in the fuzzy rule-based model. Consequently, the fuzzy rule-based model can be combined with other means of data decomposition to develop a new model.

A particularly important step in our distributed model is random sampling. In this study, sampling was random. Next, we can optimize the sampling process to further refine the model by incorporating methods such as clustering. At the stage of random sampling, we used some certain proportion to sample the features. However, with the increasing sampling percentage, the dimensionality of the data we input into the model increases. Therefore, this model can be further optimized. For example, we can reduce the number of repetitions and find the optimal parameter combination. At the same time, one may envision applying the data with similar features to a distributed model through similarity analysis. In addition, in the process of optimizing the aggregation, we can consider further optimization of the model by incorporating information granularity into the parameters of the constructed model.

Consider we have the concept of information granules, Interval-valued fuzzy sets used in our previous research. Thus, the combination of the rule-based models with information granules deserves more attention, namely granular rule-based models.

Bibliography

[1] P. Jackson, *Introduction to Expert Systems*. Washington DC: U.S. Department of Energy, 1986.

[2] R. Davis, "Expert systems: where are we? and where do we go from here?," *AI Mag*, vol. 3, no. 2, pp. 3–3, Jun. 1982.

[3] B. G. Buchanan and R. G. Smith, "Fundamentals of expert systems," *Annual Review of Computer Science*, vol. 3, no. 1, pp. 23–58, Jun. 1988.

[4] T. Takagi and M. Sugeno, "Fuzzy identification of systems and Its applications to modeling and control," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-15, no. 1, pp. 116–132, Jan. 1985.

[5] W. Pedrycz and M. Reformat, "Rule-based modeling of nonlinear relationships," *IEEE Transactions on Fuzzy Systems*, vol. 5, no. 2, pp. 256–269, May 1997.

[6] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems : Introduction and New Directions*. Cham: Springer, 2001.

[7] M. I. Rey, M. Galende, M. J. Fuente, and G. I. Sainz-Palmero, "Multi-objective based fuzzy rule based systems (FRBSs) for trade-off improvement in accuracy and interpretability: A rule relevance point of view," *Knowledge-Based Systems*, vol. 127, pp. 67–84, Jul. 2017.

[8] G. Fenza, D. Furno, V. Loia, and S. Senatore, "Approximate processing in medical diagnosis by means of deductive agents," in *The Handbook on Reasoning-Based Intelligent Systems*, World Scientific, 2013, pp. 633–657.

[9] G. Lucca *et al.*, "CC-integrals: Choquet-like Copula-based aggregation functions and its application in fuzzy rule-based classification systems," *Knowledge-Based Systems*, vol. 119, pp. 32–43, Mar. 2017.

[10] W. Pedrycz, “From fuzzy models to granular fuzzy models,” *International Journal of Computational Intelligence Systems*, vol. 9, no. 1, p. 35, Jan. 2016.

[11] E. H. Kim, S. K. Oh, and W. Pedrycz, “Reinforced rule-based fuzzy models: Design and analysis,” *Knowledge-Based Systems*, vol. 119, pp. 44–58, Mar. 2017.

[12] E. G. Mansoori, “FRBC: A fuzzy rule-based clustering algorithm,” *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 5, pp. 960–971, Oct. 2011.

[13] L. C. Dutu, G. Mauris, and P. Bolon, “A fast and accurate rule-base generation method for mamdani fuzzy systems,” *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 2, pp. 715–733, Apr. 2018.

[14] J. Kerr-Wilson and W. Pedrycz, “Generating a hierarchical fuzzy rule-based model,” *Fuzzy Sets and Systems*, vol. 381, pp. 124–139, Feb. 2020.

[15] X. Hu, Y. Shen, W. Pedrycz, X. Wang, A. Gacek, and B. Liu, “Identification of fuzzy rule-based models with collaborative fuzzy clustering,” *IEEE Transactions on Cybernetics*, doi: 10.1109/TCYB.2021.3069783.

[16] Y. Cao, Z. Zhou, C. Hu, W. He, and S. Tang, “On the interpretability of belief rule-based expert systems,” *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 11, pp. 3489–3503, Nov. 2021.

[17] N. L. Tsakiridis, J. B. Theocharis, P. Panagos, and G. C. Zalidis, “An evolutionary fuzzy rule-based system applied to the prediction of soil organic carbon from soil spectral libraries,” *Applied Soft Computing*, vol. 81, Aug. 2019, doi: https://doi.org/10.1016/j.asoc.2019.105504.

[18] M. Štěpnička and B. de Baets, “Implication-based models of monotone fuzzy rule bases,” *Fuzzy Sets and Systems*, vol. 232, pp. 134–155, Dec. 2013.

[19] N. L. Tsakiridis, J. B. Theocharis, and G. C. Zalidis, "DECO3R: A differential evolution-based algorithm for generating compact fuzzy rule-based classification systems," *Knowledge-Based Systems*, vol. 105, pp. 160–174, Aug. 2016.

[20] W. van der Aalst and E. Damiani, "Processes meet big data: Connecting data science with process science," *IEEE Transactions on Services Computing*, vol. 8, no. 6, pp. 810–819, Nov. 2015.

[21] L. Zhang, "Editorial: Big services era: Global trends of cloud computing and big data," *IEEE Transactions on Services Computing*, vol. 5, no. 4, pp. 467–468, Apr. 2012.

[22] S. Salloum, Z. Huang, and Y. He, "Random sample partition: A distributed data model for big data analysis," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 11, pp. 5846–5854, Nov. 2019.

[23] A. Segatori, F. Marcelloni, and W. Pedrycz, "On distributed fuzzy decision trees for big data," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 1, pp. 174–192, Feb. 2018.

[24] M. Mahmud, Z. Huang, S. Salloum, TamerZ. Emara, and K. Sadatdiynov, "A survey of data partitioning and sampling methods to support big data analysis," *Big Data Mining and Analytics*, vol. 3, no. 2, pp. 85–101, Jun. 2020.

[25] J. Wu, Z. Wu, J. Cao, H. Liu, G. Chen, and Y. Zhang, "Fuzzy consensus clustering with applications on big Data," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 6, pp. 1430–1445, Dec. 2017.

[26] A. Jindal, A. Dua, N. Kumar, A. Das, A. Vasilakos, and J. Rodrigues, "Providing healthcare-as-a-service using fuzzy rule based big data analytics in cloud computing," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 5, pp. 1605–1618, Sep. 2018.

[27] A. Ferranti, F. Marcelloni, A. Segatori, M. Antonelli, and P. Ducange, “A distributed approach to multi-objective evolutionary generation of fuzzy rule-based classifiers from big data,” *Information Sciences*, vol. 415–416, pp. 319–340, Nov. 2017.

[28] T. Liang and Y. Liu, “Research landscape of business intelligence and big data analytics: A bibliometrics study,” *Expert Systems with Applications*, vol. 111, pp. 2–10, Nov. 2018.

[29] Z. Zhang, P. Srivastava, D. Sharma, and P. Eachempati, “Big data analytics and machine learning: A retrospective overview and bibliometric analysis,” *Expert Systems with Applications*, vol. 184, Dec. 2021, doi: 10.1016/j.eswa.2021.115561.

[30] R. Bellman, “Dynamic programming,” *Science (1979)*, vol. 153, no. 3731, pp. 34–37, Jul. 1966.

[31] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science (1979)*, vol. 313, no. 5786, pp. 504–507, Jul. 2006, doi: 10.1126/science.1127647.

[32] G. Boquet, A. Morell, J. Serrano, and J. L. Vicario, “A variational autoencoder solution for road traffic forecasting systems: Missing data imputation, dimension reduction, model selection and anomaly detection,” *Transportation Research Part C: Emerging Technologies*, vol. 115, p. 102622, Jun. 2020, doi: 10.1016/j.trc.2020.102622.

[33] L. Liang, Y. jun Li, and S. bing Li, “Increasing the discriminatory power of DEA in the presence of the undesirable outputs and large dimensionality of data sets with PCA,” *Expert Systems with Applications*, vol. 36, no. 3 PART 2, pp. 5895–5899, 2009, doi: 10.1016/j.eswa.2008.07.022.

[34] J. C. Dunn, “A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters,” *Journal of Cybernetics*, vol. 3, no. 3, pp. 32–57, Jan. 1973.

[35] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. Boston, MA: Springer US, 1981.

[36] M. Alata, M. Molhim, and A. Ramini, "Optimizing of fuzzy c-means clustering algorithm using GA," *International Scholarly and Scientific Research & Innovation*, vol. 2, no. 3, pp. 670–675, Jan. 2008.

[37] D. C. Park and I. Dagher, "Gradient based fuzzy c-means (GBFCM) algorithm," in *IEEE International Conference on Neural Networks (ICNN'94)*, 1994, pp. 1626–1631.

[38] W. Pedrycz, "Collaborative fuzzy clustering," *Pattern Recognition Letters*, vol. 23, no. 14, pp. 1675–1686, Dec. 2002.

[39] Y. Shen and W. Pedrycz, "Collaborative fuzzy clustering algorithm: Some refinements," *International Journal of Approximate Reasoning*, vol. 86, pp. 41–61, Jul. 2017.

[40] il Hong Suh, Jae-Hyun Kim, and Frank Chung-Hoon Rhee, "Convex-set-based fuzzy clustering," *IEEE Transactions on Fuzzy Systems*, vol. 7, no. 3, pp. 271–285, Jun. 1999.

[41] H. Timm and R. Kruse, "A modification to improve possibilistic fuzzy cluster analysis," in *2002 IEEE World Congress on Computational Intelligence. 2002 IEEE International Conference on Fuzzy Systems. FUZZ-IEEE'02. Proceedings (Cat. No.02CH37291)*, pp. 1460–1465.

[42] Y. Sato, K. Izui, T. Yamada, and S. Nishiwaki, "Data mining based on clustering and association rule analysis for knowledge discovery in multiobjective topology optimization," *Expert Systems with Applications*, vol. 119, pp. 247–261, Apr. 2019.

[43] A. Onan, "Two-Stage topic extraction model for bibliometric data analysis based on word embeddings and clustering," *IEEE Access*, vol. 7, pp. 145614–145633, Oct. 2019.

[44] L. Wen, K. Zhou, and S. Yang, “A shape-based clustering method for pattern recognition of residential electricity consumption,” *Journal of Cleaner Production*, vol. 212, pp. 475–488, Mar. 2019.

[45] R. E. Precup, H. I. Filip, M. B. Rădac, E. M. Petriu, S. Preitl, and C. A. Dragoş, “Online identification of evolving Takagi–Sugeno–Kang fuzzy models for crane systems,” *Applied Soft Computing*, vol. 24, pp. 1155–1163, Nov. 2014.

[46] N. van Pham, L. T. Pham, W. Pedrycz, and L. T. Ngo, “Feature-reduction fuzzy co-clustering approach for hyper-spectral image analysis,” *Knowledge-Based Systems*, Mar. 2021, doi: https://doi.org/10.1016/j.knosys.2020.106549.

[47] M. V. Nichita, M. A. Paun, V. A. Paun, and V. P. Paun, “Image clustering algorithms to identify complicated cerebral diseases. description and comparison,” *IEEE Access*, vol. 8, pp. 88434–88442, May 2020.

[48] J. Mei, Y. Wang, L. Chen, and C. Miao, “Large scale document categorization with fuzzy clustering,” *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 5, pp. 1239–1251, Oct. 2017.

[49] S. Eschrich, Jingwei Ke, L. O. Hall, and D. B. Goldgof, “Fast accurate fuzzy clustering through data reduction,” *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 2, pp. 262–270, Apr. 2003, doi: 10.1109/TFUZZ.2003.809902.

[50] A. G. di Nuovo, M. Palesi, and V. Catania, “Multi-objective evolutionary fuzzy clustering for high-dimensional problems,” in *2007 IEEE International Fuzzy Systems Conference*, Jun. 2007, pp. 1–6. doi: 10.1109/FUZZY.2007.4295660.

[51] X. Chang, Q. Wang, Y. Liu, and Y. Wang, “Sparse regularization in fuzzy C-means for high-dimensional data clustering,” *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2616–2627, Sep. 2017.

[52] H. Zhao, J. Liang, and G. Zhang, “Fuzzy k-median clustering based on Hsim function for the high dimensional data,” in *2006 6th World Congress on Intelligent Control and Automation*, 2006, pp. 3099–3102.

[53] L. A. Zadeh, “Fuzzy sets,” *Information and Control*, vol. 8, no. 3, pp. 338–353, Jun. 1965.

[54] A. Dvořák, M. Štěpnička, and L. Štěpničková, “On redundancies in systems of fuzzy/linguistic IF–THEN rules under perception-based logical deduction inference,” *Fuzzy Sets and Systems*, vol. 277, pp. 22–43, Oct. 2015.

[55] Y. Cui, H. E, W. Pedrycz, and Z. Li, “Designing distributed fuzzy rule-based models,” *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 7, pp. 2047–2053, Jul. 2021.

[56] J. Alcala-Fdez, R. Alcala, S. Gonzalez, Y. Nojima, and S. Garcia, “Evolutionary fuzzy rule-based methods for monotonic classification,” *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 6, pp. 1376–1390, Dec. 2017.

[57] P. P. Angelov and X. Gu, “Deep rule-based classifier with human-level performance and characteristics,” *Information Sciences*, vol. 463–464, pp. 196–213, Oct. 2018.

[58] J. van der Waa, E. Nieuwburg, A. Cremers, and M. Neerincx, “Evaluating XAI: A comparison of rule-based and example-based explanations,” *Artificial Intelligence*, vol. 291, Feb. 2021, doi: 10.1016/j.artint.2020.103404.

[59] H. E, Y. Cui, W. Pedrycz, and Z. Li, “Enhancements of rule-based models through refinements of Fuzzy C-Means,” *Knowledge-Based Systems*, vol. 170, pp. 43–60, Apr. 2019.

[60] M. Antonelli, D. Bernardo, H. Hagras, and F. Marcelloni, “Multiobjective evolutionary optimization of type-2 fuzzy rule-based systems for financial data classification,” *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 2, pp. 249–264, Apr. 2017.

[61] I. ben Ali, M. Turki, J. Belhadj, and X. Roboam, "Optimized fuzzy rule-based energy management for a battery-less PV/wind-BWRO desalination system," *Energy*, vol. 159, pp. 216–228, Sep. 2018.

[62] M. Hasanipanah and H. Bakhshandeh Amnieh, "A fuzzy rule-based approach to address uncertainty in risk assessment and prediction of blast-Induced flyrock in a quarry," *Natural Resources Research*, vol. 29, no. 2, pp. 669–689, Apr. 2020.

[63] Mohd. A. Khan and A. S. Jalal, "A fuzzy rule based multimodal framework for face sketch-to-photo retrieval," *Expert Systems with Applications*, vol. 134, pp. 138–152, Nov. 2019.

[64] D. T. Bui, P. Tsangaratos, P. T. T. Ngo, T. D. Pham, and B. T. Pham, "Flash flood susceptibility modeling using an optimized fuzzy rule based feature selection technique and tree based ensemble methods," *Science of the Total Environment*, vol. 668, pp. 1038–1054, Jun. 2019.

[65] R. G. G. Caiado, L. F. Scavarda, L. O. Gavião, P. Ivson, D. L. de M. Nascimento, and J. A. Garza-Reyes, "A fuzzy rule-based industry 4.0 maturity model for operations and supply chain management," *International Journal of Production Economics*, vol. 231, Jan. 2021.

[66] W. Pedrycz and M. Reformat, "Evolutionary fuzzy modeling," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 5, pp. 652–665, Oct. 2003.

[67] B. Rezaee and M. H. F. Zarandi, "Data-driven fuzzy modeling for Takagi–Sugeno–Kang fuzzy system," *Information Sciences*, vol. 180, no. 2, pp. 241–255, Jan. 2010.

[68] S. Ye and A. Ling, "Fuzzy identification based on improved clustering arithmetic and its application," in *Proceedings - 2011 8th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2011*, 2011, vol. 1, pp. 147–151.

[69] X. Hu, W. Pedrycz, and X. Wang, "Random ensemble of fuzzy rule-based models," *Knowledge-Based Systems*, vol. 181, p. 104768, Oct. 2019.

[70] J. Kerr-Wilson and W. Pedrycz, “Some new qualitative insights into quality of fuzzy rule-based models,” *Fuzzy Sets and Systems*, vol. 307, pp. 29–49, Jan. 2017.

[71] M. A. Kacimi, O. Guenounou, L. Brikh, F. Yahiaoui, and N. Hadid, “New mixed-coding PSO algorithm for a self-adaptive and automatic learning of Mamdani fuzzy rules,” *Engineering Applications of Artificial Intelligence*, vol. 89, p. 103417, Mar. 2020.

[72] A. S. Mamaghani and W. Pedrycz, “Genetic-programming-based architecture of fuzzy modeling: towards coping with high-dimensional Data,” *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 9, pp. 2774–2784, Sep. 2021.

[73] P. Angelov, “Automatic generation of fuzzy rule-based models from data by genetic algorithms,” *Information Sciences*, vol. 150, no. 1–2, pp. 17–31, Mar. 2003.

[74] R. E. Precup, T. A. Teban, E. M. Petriu, A. Albu, and I. C. Mituletu, “Structure and evolving fuzzy models for prosthetic hand myoelectric-based control systems,” in *2018 26th Mediterranean Conference on Control and Automation (MED)*, Jun. 2018, pp. 1–9.

[75] D. D. Lee and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, no. 6755, pp. 788–791, Apr. 1999.

[76] D. D. Lee and H. S. Seung, “Algorithms for non-negative matrix factorization,” in *Advances in neural information processing systems*, 2001, vol. 13, pp. 556–562.

[77] S. Hong, J. Choi, J. Feyereisl, B. Han, and L. S. Davis, “Joint image clustering and labeling by matrix factorization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 7, pp. 1411–1424, Jul. 2016.

[78] X. Li, G. Cui, and Y. Dong, “Graph regularized non-negative low-rank matrix factorization for image vlustering,” *IEEE Transactions on Cybernetics*, vol. 47, no. 11, pp. 3840–3853, Nov. 2017.

[79] I. Kotsia, S. Zafeiriou, and I. Pitas, “A novel discriminant non-negative matrix factorization algorithm with applications to facial image characterization problems,” *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 588–595, Sep. 2007.

[80] C. W. Park, K. T. Park, and Y. S. Moon, “Eye detection using eye filter and minimisation of NMF-based reconstruction error in facial image,” *Electronics Letters*, vol. 46, no. 2, p. 130, 2010.

[81] W. Chen, Y. Zhao, B. Pan, and B. Chen, “Supervised kernel nonnegative matrix factorization for face recognition,” *Neurocomputing*, vol. 205, pp. 165–181, Sep. 2016.

[82] L. Ni, P. Manman, and W. Qiang, “Multi-Mode social network clustering via non-negative tri-matrix factorization with cluster indicator similarity regularization,” *IEEE Access*, vol. 7, pp. 151713–151723, 2019.

[83] K. W. Wilson, B. Raj, P. Smaragdis, and A. Divakaran, “Speech denoising using nonnegative matrix factorization with priors,” in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, Mar. 2008, pp. 4029–4032.

[84] M. Sun, Y. Li, J. F. Gemmeke, and X. Zhang, “Speech enhancement under low SNR conditions via noise estimation using sparse and low-rank NMF with Kullback–Leibler divergence,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 7, pp. 1233–1242, Jul. 2015.

[85] Y. Gao and G. Church, “Improving molecular cancer class discovery through sparse non-negative matrix factorization,” *Bioinformatics*, vol. 21, no. 21, pp. 3970–3975, Nov. 2005.

[86] M. M. Nazmul Arefin, “Face reconstruction using non-negative matrix factorization and ℓ<inf>1</inf> constrained optimization,” in *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)*, Jan. 2019, pp. 389–394.

[87] W. Wu, S. Kwong, J. Hou, Y. Jia, and H. H. S. Ip, "Simultaneous dimensionality reduction and classification via dual embedding regularized nonnegative matrix factorization," *IEEE Transactions on Image Processing*, vol. 28, no. 8, pp. 3836–3847, Aug. 2019.

[88] R. Huang, X. Li, and L. Zhao, "Hyperspectral unmixing based on incremental kernel nonnegative matrix factorization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 11, pp. 6645–6662, Nov. 2018.

[89] M. Gong, X. Jiang, H. Li, and K. C. Tan, "Multiobjective sparse non-negative matrix factorization," *IEEE Transactions on Cybernetics*, vol. 49, no. 8, pp. 2941–2954, Aug. 2019.

[90] L. Breiman, "Arcing classifiers," *The Annals of Statistics*, vol. 26, no. 3, pp. 801–849, Jun. 1998.

[91] M. M. Bejani and M. Ghatee, "A context aware system for driving style evaluation by an ensemble learning on smartphone sensors data," *Transportation Research Part C: Emerging Technologies*, vol. 89, pp. 303–320, Apr. 2018.

[92] J. Kerr-Wilson and W. Pedrycz, "Design of rule-based models through information granulation," *Expert Systems with Applications*, vol. 46, pp. 274–285, Mar. 2016.

[93] Y. Zhang and A. Haghani, "A gradient boosting method to improve travel time prediction," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 308–324, Sep. 2015.

[94] H. Verbois, A. Rusydi, and A. Thiery, "Probabilistic forecasting of day-ahead solar irradiance using quantile gradient boosting," *Solar Energy*, vol. 173, pp. 313–327, Oct. 2018.

[95] L. Wang, Y. Ngan, and H. Yung, "Automatic incident classification for large-scale traffic data by adaptive boosting SVM," *Information Sciences*, vol. 467, pp. 59–73, Oct. 2018.

[96] D. C. Feng *et al.*, “Machine learning-based compressive strength prediction for concrete: An adaptive boosting approach,” *Construction and Building Materials*, vol. 230, Jan. 2020, doi: 10.1016/j.conbuildmat.2019.117000.

[97] G. Hu, C. Yin, M. Wan, Y. Zhang, and Y. Fang, “Recognition of diseased Pinus trees in UAV images using deep learning and AdaBoost classifier,” *Biosystems Engineering*, vol. 194, pp. 138–151, Jun. 2020.

[98] W. Dong, Q. Yang, X. Fang, and W. Ruan, “Adaptive optimal fuzzy logic based energy management in multi-energy microgrid considering operational uncertainties,” *Applied Soft Computing*, vol. 98, Jan. 2021, doi: 10.1016/j.asoc.2020.106882.

[99] A. K. Paul, P. C. Shill, Md. R. I. Rabin, and K. Murase, “Adaptive weighted fuzzy rule-based system for the risk level assessment of heart disease,” *Applied Intelligence*, vol. 48, no. 7, pp. 1739–1756, Jul. 2018.

[100] A. Lahouar and J. ben Hadj Slama, “Hour-ahead wind power forecast based on random forests,” *Renewable Energy*, vol. 109, pp. 529–541, Aug. 2017.

[101] L. F. Santos Pereira, S. Barbon, N. A. Valous, and D. F. Barbin, “Predicting the ripening of papaya fruit with digital imaging and random forests,” *Computers and Electronics in Agriculture*, vol. 145, pp. 76–82, Feb. 2018.

[102] J. H. Friedman, “Greedy function approximation: a gradient boosting machines,” *Ann Stat*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.

[103] T. K. Ho, “The random subspace method for constructing decision forests,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[104] B. V. Dasarathy and B. V. Sheela, “A composite classifier system design: Concepts and methodology,” *Proceedings of the IEEE*, vol. 67, no. 5, pp. 708–713, 1979.

[105] Y. Freund and R. E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, Aug. 1997.

[106] Q. Zhang, W. Hu, Z. Liu, and J. Tan, “TBM performance prediction with bayesian optimization and automated machine learning,” *Tunnelling and Underground Space Technology*, vol. 103, Sep. 2020.

[107] P. Kumari and D. Toshniwal, “Extreme gradient boosting and deep neural network based ensemble learning approach to forecast hourly solar irradiance,” *Journal of Cleaner Production*, vol. 279, Jan. 2021, doi: 10.1016/j.jclepro.2020.123285.

[108] T. Zhang, W. He, H. Zheng, Y. Cui, H. Song, and S. Fu, “Satellite-based ground PM2.5 estimation using a gradient boosting decision tree,” *Chemosphere*, vol. 268, Apr. 2021, doi: 10.1016/j.chemosphere.2020.128801.

[109] A. Cauchy, “ Méthode générale pour la résolution des systemes d’équations simultanées,” *Comp. Rend. Sci. Paris*, vol. 25, pp. 536–538, 1847.

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

[111] D. Wu, Y. Yuan, J. Huang, and Y. Tan, “Optimize TSK fuzzy systems for regression problems: minibatch gradient descent with regularization, droprule, and AdaBound (MBGD-RDA),” *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 5, pp. 1003–1015, May 2020.

[112] M. Moradi Fard, T. Thonet, and E. Gaussier, “Deep k-Means: Jointly clustering with k-Means and learning representations,” *Pattern Recognition Letters*, vol. 138, pp. 185–192, Oct. 2020.

[113] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv*, vol. 1412, no. 6980, Dec. 2014.

[114] S. Mukherjee, P. Niyogi, T. Poggio, and R. Rifkin, “Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization,” *Advances in Computational Mathematics*, vol. 25, no. 1–3, pp. 161–193, Jul. 2006.

[115] M. Stone, “Cross-validation:a review,” *Series Statistics*, vol. 9, no. 1, pp. 127–139, Jan. 1978.

[116] M. W. Browne, “Cross-validation methods,” *Journal of Mathematical Psychology*, vol. 44, no. 1, pp. 108–132, Mar. 2000.

[117] S. Makridakis, “Accuracy measures: theoretical and practical concerns,” *International Journal of Forecasting*, vol. 9, no. 4, pp. 527–529, Dec. 1993.

[118] L. A. Zadeh, “Fuzzy sets and information granularity,” *Advances in Fuzzy Set Theory and Applications*, vol. 11, pp. 3–18, 1979.

[119] A. Bargiela and W. Pedrycz, *Granular Computing*. Boston, MA: Springer US, 2003.

[120] A. A. Ramli, J. Watada, and W. Pedrycz, “Information granules problem: An efficient solution of real-time fuzzy regression analysis,” 2015, pp. 39–61.

[121] W. Pedrycz, R. Al-Hmouz, A. Morfeq, and A. S. Balamash, “Building granular fuzzy decision support systems,” *Knowledge-Based Systems*, vol. 58, pp. 3–10, Mar. 2014.

[122] X. Hu, W. Pedrycz, and X. Wang, “From fuzzy rule-based models to their granular generalizations,” *Knowledge-Based Systems*, vol. 124, pp. 133–143, May 2017.

[123] W. Pedrycz, “The principle of justifiable granularity and an optimization of information granularity allocation as fundamentals of granular computing,” *Journal of Information Processing Systems*, vol. 7, no. 3, pp. 397–412, Sep. 2011.

[124] J. T. Yao, A. v. Vasilakos, and W. Pedrycz, “Granular computing: Perspectives and challenges,” *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1977–1989, Dec. 2013.

[125] W. Pedrycz and W. Homenda, “Building the fundamentals of granular computing: A principle of justifiable granularity,” *Applied Soft Computing*, vol. 13, no. 10, pp. 4209–4218, Oct. 2013.

[126] D. Wang and J. Gu, “VASC: Dimension reduction and visualization of single-cell RNA-seq data by deep variational autoencoder,” *Genomics, Proteomics and Bioinformatics*, vol. 16, no. 5, pp. 320–331, Oct. 2018.

[127] F. J. Pulgar, F. Charte, A. J. Rivera, and M. J. del Jesus, “Choosing the proper autoencoder for feature fusion based on data complexity and classifiers: Analysis, tips and guidelines,” *Information Fusion*, vol. 54, pp. 44–60, Feb. 2020.

[128] G. Boquet, A. Morell, J. Serrano, and J. L. Vicario, “A variational autoencoder solution for road traffic forecasting systems: Missing data imputation, dimension reduction, model selection and anomaly detection,” *Transportation Research Part C: Emerging Technologies*, vol. 115, Jun. 2020, doi: 10.1016/j.trc.2020.102622.

[129] J. Zhao *et al.*, “Attribute mapping and autoencoder neural network based matrix factorization initialization for recommendation systems,” *Knowledge-Based Systems*, vol. 166, pp. 132–139, Feb. 2019, doi: 10.1016/j.knosys.2018.12.022.

[130] S. Ryu, H. Choi, H. Lee, and H. Kim, “Convolutional autoencoder based feature extraction and clustering for customer load analysis,” *IEEE Transactions on Power Systems*, vol. 35, no. 2, pp. 1048–1060, Mar. 2020.

[131] H. E, Y. Cui, W. Pedrycz, and Z. Li, “Fuzzy relational matrix factorization and its granular characterization in data description,” *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 3, pp. 794–804, Mar. 2022, [Online]. Available: https://ieeexplore.ieee.org/document/9311791/

[132] Y. Yang, A. Ming, Y. Zhang, and Y. Zhu, “Discriminative non-negative matrix factorization (DNMF) and its application to the fault diagnosis of diesel engine,” *Mechanical Systems and Signal Processing*, vol. 95, pp. 158–171, Oct. 2017.

[133] L. Zhang, L. Zhang, B. Du, J. You, and D. Tao, “Hyperspectral image unsupervised classification by robust manifold matrix factorization,” *Information Sciences*, vol. 485, pp. 154–169, Jun. 2019, doi: 10.1016/j.ins.2019.02.008.

[134] Z. Li, J. Tang, and X. He, “Robust structured nonnegative matrix factorization for image representation,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 5, pp. 1947–1960, May 2018, doi: 10.1109/TNNLS.2017.2691725.

[135] X. Kang, X. Xiang, S. Li, and J. A. Benediktsson, “PCA-based edge-preserving features for hyperspectral image classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 12, pp. 7140–7151, Dec. 2017.

[136] C. Happ and S. Greven, “Multivariate functional principal component analysis for data observed on different (dimensional) domains,” *J Am Stat Assoc*, vol. 113, no. 522, pp. 649–659, Apr. 2018.

[137] S. Yi, Z. Lai, Z. He, Y. ming Cheung, and Y. Liu, “Joint sparse principal component analysis,” *Pattern Recognition*, vol. 61, pp. 524–536, Jan. 2017.

[138] M. M. Nazmul Arefin, “Face reconstruction using non-negative matrix factorization and ℓ1 constrained optimization,” in *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)*, Jan. 2019, pp. 389–394.

[139] T. M. Oshiro, P. S. Perez, and J. A. Baranauskas, “How many trees in a random forest?,” in *Machine Learning and Data Mining in Pattern Recognition*, 2012, pp. 154–168.

[140] W. Pedrycz, V. Loia, and S. Senatore, “Fuzzy clustering with viewpoints,” *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 2, pp. 274–284, Apr. 2010.

[141] X. Hu, W. Pedrycz, G. Wu, and X. Wang, “Data reconstruction with information granules: An augmented method of fuzzy clustering,” *Applied Soft Computing*, vol. 55, pp. 523–532, Jun. 2017.

[142] W. Pedrycz, R. Al-Hmouz, A. S. Balamash, and A. Morfeq, “Designing granular fuzzy models: A hierarchical approach to fuzzy modeling,” *Knowledge-Based Systems*, vol. 76, pp. 42–52, Mar. 2015.

[143] V. J. Kok and C. S. Chan, “GrCS: Granular computing-based crowd segmentation,” *IEEE Transactions on Cybernetics*, vol. 47, no. 5, pp. 1157–1168, May 2017.

[144] M. Sugeno and T. Yasukawa, “A fuzzy-logic-based approach to qualitative modeling,” *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 1, p. 7, Feb. 1993.

[145] A. Bargiela and W. Pedrycz, “Granular computing,” in *Handbook on Computational Intelligence*, World Scientific, 2016, pp. 43–66.

[146] H. Fujita, A. Gaeta, V. Loia, and F. Orciuoli, “Improving awareness in early stages of security analysis: A zone partition method based on GrC,” *Applied Intelligence*, vol. 49, no. 3, pp. 1063–1077, Mar. 2019.

[147] H. Fujita, A. Gaeta, V. Loia, and F. Orciuoli, “Resilience analysis of critical infrastructures: a cognitive approach based on granular computing,” *IEEE Transactions on Cybernetics*, vol. 49, no. 5, pp. 1835–1848, May 2019.