Soft Comput (2018) 22:3155–3170
https://doi.org/10.1007/s00500-017-2925-8

FOCUS

An extended Takagi–Sugeno–Kang inference system (TSK+) with fuzzy interpolation and its rule base generation

Jie Li[1] · **Longzhi Yang**[1] · **Yanpeng Qu**[2] · **Graham Sexton**[1]

Published online: 20 November 2017

Abstract A rule base covering the entire input domain is required for the conventional Mamdani inference and Takagi–Sugeno–Kang (TSK) inference. Fuzzy interpolation enhances conventional fuzzy rule inference systems by allowing the use of sparse rule bases by which certain inputs are not covered. Given that almost all of the existing fuzzy interpolation approaches were developed to support the Mamdani inference, this paper presents a novel fuzzy interpolation approach that extends the TSK inference. This paper also proposes a data-driven rule base generation method to support the extended TSK inference system. The proposed system enhances the conventional TSK inference in two ways: (1) workable with incomplete or unevenly distributed data sets or incomplete expert knowledge that entails only a sparse rule base and (2) simplifying complex fuzzy inference systems by using more compact rule bases for complex systems without the sacrificing of system performance. The experimentation shows that the proposed system overall outperforms the existing approaches with the utilisation of smaller rule bases.

Keywords Fuzzy inference system · TSK · Fuzzy rule base generation · Fuzzy interpolation

Communicated by P. Angelov, F. Chao.

✉ Longzhi Yang
longzhi.yang@northumbria.ac.uk

[1] Deapartment of Computer and Information Science, Northumbria University, Newcastle upon Tyne NE1 8ST, UK

[2] Information Science and Technology College, Dalian Maritime University, Dalian, People's Republic of China

1 Introduction

Fuzzy inference mechanisms are built upon fuzzy logic to map system inputs and outputs. A typical fuzzy inference system consists of a rule base and an inference engine. A number of inference engines have been developed, with the Mamdani inference (Mamdani (1977)) and the TSK inference (Takagi and Sugeno (1985)) being the most widely used. The Mamdani fuzzy model is more intuitive and suitable for handling linguistic inputs; its outputs are usually fuzzy sets, and thus, a defuzzification process is often required. In contrast, the TSK inference approach produces crisp outputs directly, as TSK fuzzy rules use polynomials as rule consequences. There are generally two types of rule bases used to support the two fuzzy inference engines, which are Mamdani-style rule bases and TSK-style rule bases accordingly.

A rule base, Mamdani-style or TSK-style, can either be translated from expert knowledge or extracted from data. The rule base led by the knowledge-driven approaches therefore essentially is a representation of the human experts' knowledge in the format of fuzzy rules (Negnevitsky (2005)). In order to enable this approach, a problem has to be human comprehensible and fully understood by human experts as linguistic rules, which are then interpreted as fuzzy rules by specifying the membership functions of linguistic words. Recognising that the expert knowledge may not always be available, data-driven approaches were proposed, which extract fuzzy rules from a set of training data using machine learning approaches (Rezaee and Zarandi (2010)). Both Mamdani and TSK inference approaches are only workable with dense rule bases which each covers the entire input domain.

Fuzzy interpolation relaxes the requirement of dense rule bases from conventional fuzzy inference systems (Kóczy and Hirota (1993); Yang et al. (2017c)). When a given

observation does not overlap with any rule antecedent, certain conclusions can still be generated by means of interpolation. In addition, fuzzy interpolation helps in system complexity reduction by removing the rules that can be approximated by their neighbours. A number of fuzzy interpolation approaches have been developed, such as Chen and Hsin (2015), Huang and Shen (2006, 2008), Kóczy and Hirota (1997), Naik et al. (2014), Yang and Shen (2011), Yang et al. (2017a), Shen and Yang (2011) and Yang and Shen (2013). However, all these existing fuzzy interpolation approaches were extensions of the Mamdani inference.

A novel fuzzy interpolation approach, which extends the TSK inference, is presented in this paper. The proposed fuzzy inference engine is workable with sparse, dense or imbalanced TSK-style rule bases, which is a further development on the seminal work of Li et al. (2017). In addition, a data-driven TSK-style rule base generation approach is also proposed to extract compact and concise rule bases from incomplete, imbalanced, normal or over-dense data sets. Note that sparse and imbalanced data sets are still commonly seen, regardless of the magnitude of the data sets in the era of big data. The proposed approach has been applied to two benchmark problems and a real-world problem in the field of cyber security. The experimentation demonstrated the power of the proposed approach in enhancing the conventional TSK inference method by means of broader applicability and better system efficiency, and competitive performance in reference to other machine learning approaches.

The structure of rest of the paper is organised as follows. Section 2 introduces the theoretical underpinnings of TSK fuzzy inference model and the TSK rule base generation approaches. Section 3 presents the extended TSK system, and Sect. 4 discusses the proposed rule base generation approach. Section 5 details the experimentation for demonstration and validation. Section 6 concludes the paper and suggests probable future developments.

2 Background

The conventional TSK inference system and rule base generation approaches are briefly reviewed in this section.

2.1 TSK inference

Suppose a TSK-style fuzzy rule base comprises of n rules each with m antecedents:

$$\begin{aligned} R_1: \ &\textbf{IF } x_1 \text{ is } A_{11} \text{ and } \ldots \text{and } x_m \text{ is } A_{m1} \\ &\textbf{THEN } y = f_1(x_1, \ldots, x_m) \\ &\qquad = \beta_{01} + \beta_{11}x_1 + \ldots + \beta_{m1}x_m, \\ &\cdots \\ R_n: \ &\textbf{IF } x_1 \text{ is } A_{1n} \text{ and } \ldots \text{and } x_m \text{ is } A_{mn} \\ &\textbf{THEN } y = f_n(x_1, \ldots, x_m) \\ &\qquad = \beta_{0n} + \beta_{1n}x_1 + \ldots + \beta_{mn}x_m, \end{aligned} \tag{1}$$

where β_{0r} and β_{sr}, ($r \in \{1, 2, \ldots, n\}$ and $s \in \{1, 2, \ldots, m\}$) are constant parameters of the linear functions of rule consequences. The consequence polynomials deteriorate to constant numbers β_{0r} when the outputs are discrete crisp numbers (to represent symbolic values). Given an input vector $(A_1^*, \ldots, A_m^*)$, the TSK engine performs inference in the following steps:

1 Determine the firing strength of each rule R_r ($r \in \{1, 2, \ldots, n\}$) by integrating the similarity degrees between its antecedents and the given inputs:

$$\alpha_r = S(A_1^*, A_{1r}) \wedge \ldots \wedge S(A_m^*, A_{mr}), \tag{2}$$

where $\wedge$ is a t-norm usually implemented as a minimum operator, and $S(A_s^*, A_{sr})$ ($s \in \{1, 2, \ldots, m\}$) is the similarity degree between fuzzy sets A_s^* and A_{sr}:

$$S(A_s^*, A_{sr}) = max\{min\{\mu_{A_s^*}(x), \mu_{A_{sr}}(x)\}\}, \tag{3}$$

where $\mu_{A_s^*}(x)$ and $\mu_{A_{sr}}(x)$ are the degrees of membership for a given value x within the domain.

2 Calculate the sub-output led by each rule R_r based on the given observation $(A_1^*, \ldots, A_m^*)$:

$$\begin{aligned} &f_r(x_1^*, \ldots, x_m^*) \\ &\quad = \beta_{0r} + \beta_{1r} Rep(A_1^*) + \cdots + \beta_{mr} Rep(A_m^*), \end{aligned} \tag{4}$$

where $Rep(A_s^*)$ is the representative value or defuzzified value of fuzzy set fuzzy set A_s^*, which is often calculated as the centre of area of the membership function.

3 Generate the final output by integrating all the sub-outputs from all the rules:

$$y = \frac{\sum_{r=1}^{n} \alpha_r f_r(x_1^*, \ldots, x_m^*)}{\sum_{r=1}^{n} \alpha_r}. \tag{5}$$

It is clear from Eq. 3 that the firing strength will be 0 if a given input vector does not overlap with any rule antecedent. In this case, no rule will be fired and the conventional TSK approach will fail.

2.2 TSK rule base generation

The antecedent variables of a TSK rule are represented as fuzzy sets and the consequence is represented by a linear polynomial function, as shown in Eq. 1. Data-driven fuzzy

rule extraction approaches first partition the problem domain into multiple regions or rule clusters. Then, each region is represented by a TSK rule. Various clustering algorithms, such as K-Means (MacQueen (1967)) and its variations, can be used to divide the problem domain into sub-regions or rule clusters (Chen and Linkens (2004); Rezaee and Zarandi (2010)).

Given a rule cluster that contains a set of multi-dimensional data instances, a typical TSK fuzzy rule extraction process is performed in two steps: (1) rule antecedents determination and (2) consequent polynomial determination (Nandi and Klawonn (2007)). The rule antecedent determination process prescribes a fuzzy set to represent the information of the cluster on each dimension, such as Gaussian membership functions Chen and Linkens (2004). The consequent polynomial can usually be determined by employing the linear regression (Nandi and Klawonn (2007); Rezaee and Zarandi (2010)). Once each rule cluster has been expressed by a TSK rule, the TSK rule base can be assembled by combining all the extracted rules.

Note that a 0-order TSK fuzzy model is required if only symbolic labels are included in the output of data set (Kerk et al. (2016)). In this case, the step of consequent polynomial determination is usually omitted. Instead, discrete crisp numbers are typically used in representing the symbolic output values. Accordingly, the rule base generation process is different by firstly dividing the labelled data set into multiple sub-data sets each sharing the same label. Then, a clustering algorithm is applied to each sub-data set to generate rule clusters. Finally, each rule cluster is represented as the antecedents of a rule with an integer number used as the 0-order consequence representing the label.

3 TSK inference with fuzzy interpolation (TSK+)

The conventional TSK fuzzy inference system is extended in this section by allowing the interpolation and extrapolation of inference results. The extended system is thus workable with sparse rule bases, dense rule bases and imbalanced rule bases, which is termed as TSK+ inference system.

3.1 Modified similarity measure

Conventional TSK will fail if a given input does not overlap any rule antecedent in the rule base. This can be addressed using fuzzy interpolation such that the inference consequence can be approximated from the neighbouring rules of the given input. In order to enable this, the measure of firing strength used in the conventional TSK inference is modified based on a revised similarity measure proposed in Chen and Chen (2003). In particular, the similarity measure proposed in Chen and Chen (2003) is not sensitive to distance in addition to membership functions. This similarity measure is further extended in this subsection such that its sensitivity to distance is flexible and configurable to support the development of TSK+ inference engine.

It has been proven in the literature that different types of membership functions do not pose a significant difference in inference results if the membership functions are properly fine-tuned (Chang and Fan (2008)). Based on this, only triangular membership functions are used in this work for computational efficiency. Given two triangular fuzzy sets $A = (a_1, a_2, a_3)$ and $A' = (a'_1, a'_2, a'_3)$ in a normalised variable domain, their similarity degree $S(A, A')$ can be calculated as (Chen and Chen (2003)):

$$S(A, A') = \left(1 - \frac{\sum_{i=1}^{3} |a_i - a'_i|}{3}\right). \tag{6}$$

Equation 6 is extended in this work by introducing a configurable parameter as:

$$S(A, A') = \left(1 - \frac{\sum_{i=1}^{3} |a_i - a'_i|}{3}\right) \cdot d, \tag{7}$$

where d, termed as $distance\ factor$, is a function of the distance between the two concerned fuzzy sets:

$$d = \begin{cases} 1 & ; \ a_1 = a_2 = a_3 \\ & \ \ \& \ a'_1 = a'_2 = a'_3 \\ 1 - \frac{1}{1+e^{(-s \cdot \|A, A'\| + 5)}} & ; \quad \text{otherwise}, \end{cases} \tag{8}$$

where $\|A, A'\|$ represents the distance between the two fuzzy sets usually defined as the Euclidean distance of their representative values, and s ($s > 0$) is an adjustable sensitivity factor. Smaller value of s leads to a similarity degree which is more sensitive to the distance of the two fuzzy sets. The constant 5 in the equation ensures that the distance factor is normalised as 1 when the distance between two given fuzzy sets is 0 (i.e. the two fuzzy sets have the same representative values). According to Eq. 8, the *distance factor* is not considered when fuzzy sets A and A' are both crisp. This is because the shapes of the fuzzy set

need to be considered by the representative values as contributing elements of the *distance factor* when the objects are fuzzy sets, but there is no point to consider this element if the objects are crisp numbers (Johanyák and Kovács (2005)).

The modified similarity measure $S(A, A')$ between fuzzy sets A and A' has the following properties:

1. lager value of $S(A, A')$ represents higher similarity degree between fuzzy sets A and A';
2. $S(A, A') = 1$ if and only if fuzzy sets A and A' are identical;
3. $S(A, A') > 0$ unless ($a_1 = a_2 = a_3 = 0$ and $a'_1 = a'_2 = a'_3 = 1$) or ($a_1 = a_2 = a_3 = 1$ and $a'_1 = a'_2 = a'_3 = 0$).

3.2 Extended TSK inference

Given a rule base as specified in Eq. 1 and an input vector $(A_1^*, \ldots, A_m^*)$, the TSK+ performs inferences using the same steps as those detailed in Sect. 2.1 except that Eq. 3 is replaced by Eq. 7. According to the third property of the modified similarity measure discussed above, $S(A_s^*, A_{sr}) > 0$ unless A_s^* and A_{sr} take boundary crisp values 0 and 1. This means the firing strength of any rule R_r is always greater than 0, i.e. $\alpha_r > 0$, except for the special case when only boundary crisp values are involved. As a result, every rule in the rule base contributes to the final inference result to a certain degree. Therefore, even if the given observation does not overlap with any rule antecedent in the rule base, certain inference result can still be generated, which significantly improves the applicability of the conventional TSK inference system.

4 Sparse TSK rule base generation

A data-driven TSK-style rule base generation approach for the proposed TSK+ inference engine is presented in this section, which is outlined in Fig. 1. Given a data set $\mathbb{T}$ which might be sparse, unevenly distributed, or dense, the system firstly groups the data instances into clusters using certain clustering algorithms. Then, each cluster is expressed as a TSK rule by employing linear regression. From this, an initial rule base is generated by combining all the extracted rules. Finally, the initialised rule base is optimised by applying the genetic algorithm (GA), which fine-tunes the membership functions of fuzzy sets in the rule base.

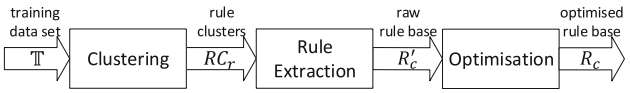

Fig. 1 TSK+ rule base generation

4.1 Rule base initialisation

Centroid-based clustering algorithms are traditionally employed in TSK fuzzy modelling to group similar objects together for rule extraction (Chen and Linkens (2004)), which is also the case in this work. However, differing from existing TSK-style rule base generation approaches, the proposed system is workable with dense data sets, sparse data sets and unevenly distributed data sets. Therefore, a two-level clustering scheme is applied in this work. The first level of clustering divides the given (dense/sparse) data set into multiple sub-data sets using sparse K-Means clustering algorithm (Witten and Tibshirani (2010)). Based on the feature of the sparse K-Means clustering, those divided sub-data sets are generally considered being dense. The second level of clustering is applied on each obtained dense sub-data set to generate rule clusters for TSK fuzzy rule extraction by employing the standard K-Means clustering algorithm (MacQueen (1967)). Note that the number of clusters has to be pre-defined for both sparse K-Means and the standard K-Means, which is discussed first below.

4.1.1 Number of clusters determination

A number of approaches have been proposed in the literature to determine the value of k, such as the Elbow method, Cross-validation, Bayesian Information Criterion (BIC)-based approach, and Silhouette-based approach (Kodinariya and Makwana (2013)). In particular, the Elbow method is faster and effective, and this approach is therefore employed in this work. This approach determines the number of clusters based on the criteria that adding another cluster does not lead to much better modelling result based on a given objective function. For instance, for a given problem, the relationship between performance improvement and the value of k is shown in Fig. 2. The value of k in this case can be determined as 4 which is the obvious turning point (or the Elbow point).

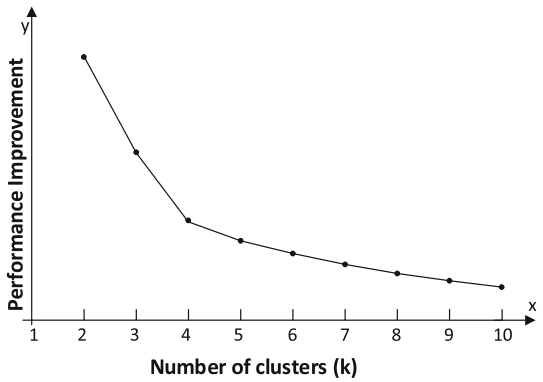

Fig. 2 Determination of k using the Elbow method

4.1.2 Dense sub-data set generation

Sparse K-Means is an extension of the standard K-Means for handling sparse data sets (Witten and Tibshirani (2010)). Assuming k clusters are required, sparse K-Means also starts with the initialisation of k centroids (usually randomly). This is followed by the assignment of data instances to centroids and the updating of centroids based on the assignments, which are iterated until there is no change on the assignments. Different from the standard K-Means which assigns objects with the goal of minimising the within-cluster sum of squares error (SSE), the sparse K-Means assigns objects by maximising the between-cluster sum of squares error (BCSS), which is defined as:

$$BCSS = \sum_{q=1}^{m+1} \left(\sum_{t=1}^{p} (x_{tq} - \mu_q)^2 - SSE \right), \tag{9}$$

where p is the total numbers of data instances in the given data set, m is the number of input features in the given data set, μ_q is the mean of all the elements on the qth feature, and x_{tq} is the qth feature of the tth data point in the given data set.

The within-cluster sum of squares error SSE is defined as:

$$SSE = \sum_{j=1}^{k} \sum_{t=1}^{p_j} \left(\parallel x_{jt} - v_j \parallel \right)^2, \tag{10}$$

where k is the number of clusters determined by the Elbow approach, p_j is the number of data instances in the jth cluster, x_{jt} is the tth data point in the jth cluster, v_j is the jth cluster centre, and $\parallel x_{jt} - v_j \parallel$ is the Euclidean distance between x_{jt} and v_j. Note that if the labels in a given data set are symbolic values, only 0-order TSK rules are required and thus Eq. 9 becomes:

$$BCSS = \sum_{q=1}^{m} \left(\sum_{t=1}^{p} (x_{tq} - \mu_q)^2 - SSE \right). \tag{11}$$

4.1.3 Rule cluster generation

Once the given training data set $\mathbb{T}$ has been divided into k dense sub-data sets, K-Means is employed to each determined sub-data set T_i $(1 \leq i \leq k)$ to generate rule clusters, each representing a rule. Assume that k_i clusters are required for a sub-data set T_i. K-Means is initialised by k_i random cluster centroids. It then assigns every data instance to one cluster by minimising the SSE:

$$SSE = \sum_{j=1}^{k_i} \sum_{t=1}^{p_t^i} \left(\parallel x_{jt}^i - v_j^i \parallel \right)^2, \tag{12}$$

where p_t^i is the number of data points in the jth cluster of the sub-data set T_i, x_{jt}^i is the tth data point in the jth cluster in the sub-data set T_i, v_j^i is the centre of the jth cluster in the sub-data set T_i, and $\parallel x_{jt}^i - v_j^i \parallel$ is the Euclidean distance between x_{jt}^i and v_j^i. Once all the data instances are assigned, the algorithm updates the cluster centroids accordingly to the newly assigned members. These two steps are iterated until there is no change in object assignments. After the K-Means is applied, the given training data set $\mathbb{T}$ is divided into $n = \sum_{j=1}^{k} k_i$ clusters. For simplicity, the generated rule clusters are jointly represented as $\{RC_1, RC_2, \ldots, RC_n\}$.

4.1.4 Fuzzy rule extraction

Each determined cluster from the above steps is utilised to form one TSK fuzzy rule. A number of approaches have been proposed to use a Gaussian membership function to represent a cluster, such as Rezaee and Zarandi (2010). However, given the fact that most of the real-world data are not normally distributed, the cluster centroid usually is not identical with the centre of the Gaussian membership function, and thus, Gaussian membership functions may not be able to accurately represent the distribution of the calculated clusters. In order to prevent this and also keep computational efficiency as stated in Sect. 3, triangular membership functions are utilised in this work.

Suppose that a data set has m input features and a single-output feature. Given a rule cluster RC_r $(1 \leq r \leq n)$, a TSK fuzzy rule R_r can be extracted from the cluster as follows:

$$\begin{aligned} R_r : \ & \textbf{IF } x_1 \text{ is } A_{1r} \text{ and } \ldots \text{ and } x_m \ is \ A_{mr} \\ & \textbf{THEN } y = f_r(x_1, \ldots, x_m). \end{aligned} \tag{13}$$

Without loss generality, take the sth dimension $(1 \leq s \leq m)$ of rule cluster RC_r as an example, denoted as RC_r^s. Suppose that RC_r^s has p_r elements, i.e. $RC_r^s = \{x_s^1, x_s^2, \ldots, x_s^{p_r}\}$. As only triangular fuzzy sets are used in this work, fuzzy set A_{sr} can be precisely represented as $(a_{sr}^1, a_{sr}^2, a_{sr}^3)$. The core of the triangular fuzzy set is set as the cluster centroid, that is $a_{sr}^2 = \sum_{q=1}^{p_r} x_s^q / p_r$; and the support of the fuzzy set is set as the span of the cluster, i.e. $(a_{sr}^1, a_{sr}^3) = (min\{x_s^1, x_s^2, \ldots, x_s^{p_r}\}, max\{x_s^1, x_s^2, \ldots, x_s^{p_r}\})$.

First-order polynomials are typically used as the consequences of TSK fuzzy rules. That is, $y = \beta_{0r} + \beta_{1r} x_1 + \cdots + \beta_{mr} x_m$, where the parameters β_{0r} and β_{sr}, $s \in \{1, 2, \ldots, m\}$ are estimated using a linear regression approach. The locally weighted regression (LWR) is particularly adopted in this work, due to its ability to generate an independent model that is only related to the given cluster of data in the training data set (Nandi and Klawonn (2007); Rezaee and Zarandi (2010)). The rule consequence will deteriorate to 0-order, if the values in the output dimension are discrete integer num-

bers. From this, the raw base is initialised by combining all the extracted rules, which is of the form of Eq. 1.

4.2 Rule base optimisation

The generated raw rule base is optimised in this section by fine-tuning the membership functions using the general optimisation searching algorithm, genetic algorithm (GA). GA has been successfully utilised in rule base optimisation, such as Mucientes et al. (2009) and Tan et al. (2016). Briefly, GA is an adaptive heuristic search algorithm for solving both constrained and unconstrained optimisation problems based on evolutionary ideas of natural selection process that mimics biological evolution. The algorithm firstly initialises the population with random individuals. It then selects a number of individuals for reproduction by applying the genetic operators. The offspring and some of the selected existing individuals jointly form the next generation. The algorithm repeats this process until a satisfactory solution is generated or a maximum number of generations has been reached.

4.2.1 Problem representation

Assume that an initialised TSK rule base is comprised of n rules as expressed in Eq. 1. A chromosome or individual, denoted as I, in the GA is used to represent a potential solution, which is designed to represent the rule base in this proposed system, as illustrated in Fig. 3.

4.2.2 Population initialisation

The initial population $\mathbb{P} = \{I_1, I_2, \ldots, I_{|\mathbb{P}|}\}$ is formed by taking the initialised rule base and its random variations. In order to guarantee all the variated fuzzy sets are valid and convex, constraint $a_{sr}^1 < a_{sr}^2 < a_{sr}^3$ is applied to the genes representing each fuzzy set. The size of the population $|\mathbb{P}|$ is a problem-specific adjustable parameter, typically ranging from tens to thousands, with 20–30 being used most commonly (Naik et al. (2014)).

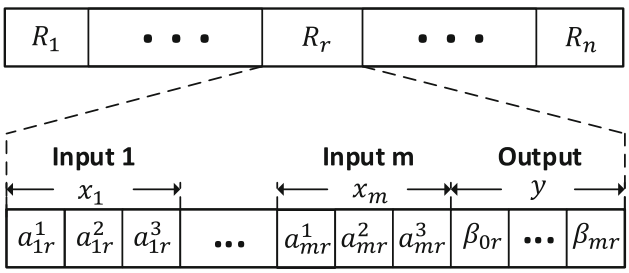

Fig. 3 Chromosome encoding

4.2.3 Objective function

An objective function is used in the GA to determine the quality of individuals. The objective function in this work is defined as the root mean square error (RMSE). Given a training data set $\mathbb{T}$ and an individual I_i, $1 \leq i \leq |\mathbb{P}|$, the RMSE value can be calculated as:

$$RMSE_i = \sqrt{\frac{\sum_{j=1}^{|\mathbb{T}|} (z_j - \hat{z}_j)^2}{|\mathbb{T}|}}, \tag{14}$$

where $|\mathbb{T}|$ is the size of the given training data set, z_j is the label of the jth training data instance, and $\hat{z}_j$ represents the output value led by the proposed TSK+ inference approach. The individual with the smallest value of RMSE represents the fittest solution in the population.

4.2.4 Selection

A number of individuals need to be selected for reproduction, which is implemented in this work by the fitness proportionate selection method, also known as the roulette wheel selection. Assuming that f_i is the fitness of individual I_i in the current population $\mathbb{P}$, its probability of being selected to generate the next generation is:

$$p(I_i) = \frac{f_i}{\sum_{j=1}^{|\mathbb{P}|} f_j}, \tag{15}$$

where $|\mathbb{P}|$ is the size of the population. The fitness value f_i of an individual I_i in the proposed system was determined by adopting the linear-ranking algorithm (Baker (1985)) given as:

$$f_i = 2 - max + \frac{2(max - 1)(r_i - 1)}{|\mathbb{P}|}, \tag{16}$$

where r_i is the ranking position of individual I_i in the ordered population $\mathbb{P}$, and max is the bias or selective pressure towards the fittest individuals in the population.

4.2.5 Reproduction

Once a number of parents are selected, they then breed some individuals for the next generation using the genetic operators crossover and mutation, as shown in Fig. 4. In particular, crossover swaps contiguous parts of the genes of two individuals. In this work, the single-point crossover approach is adopted, which swaps all data beyond this index point between the two parent chromosomes to generate two children. Note that the crossover point can only be between those genes which employed to represent two different fuzzy sets,

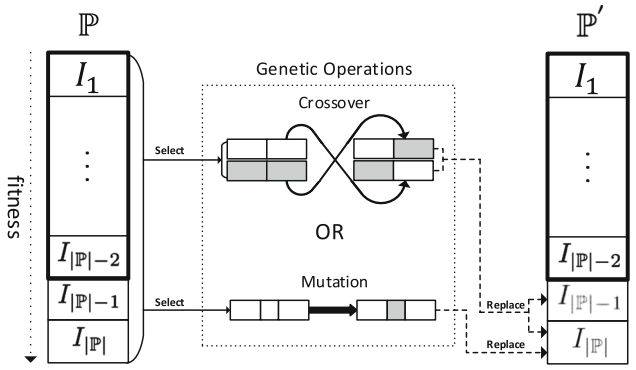

Fig. 4 Procedure of reproduction (with only one crossover or mutation operation per generation for illustration)

such that all the fuzzy sets are valid and convex all the time during the reproduction process.

The second genetic operator mutation is used to maintain genetic diversity from one generation of an individual to the next, which is analogous to a biological mutation. Mutation alters one gene values in a chromosome from its initial state. A pre-defined mutation rate is used to control the percentage of occurrence of mutations. In this work, in order to make sure the resulted fuzzy sets are valid and convex, the constraint $a^1_{sr} \geq a^2_{sr} \geq a^3_{sr}$ is applied to the genes representing each fuzzy set during the mutation operation. The newly bred individuals and some of the best individual in the current generation $\mathbb{P}$ jointly form the next generation of the population ($\mathbb{P}'$).

4.2.6 Iteration and termination

The selection and reproduction processes are iterated until the pre-defined maximum number of iterations is reached or the value of the objective function regarding an individual is less than a pre-specified threshold. When the termination condition is satisfied, the fittest individual in the current population is the optimal solution.

5 Experimentation

Two nonlinear mathematical models and a well-known real-world data set (KDD Cup 99 data set) are employed in this section to validate and evaluate the proposed system.

5.1 Experiment 1

A 3-dimensional nonlinear function has been used as a benchmark by a number of projects, including the recent ones such as Bellaaj et al. (2013); Tan et al. (2016) and Li et al. (2017), which is re-considered in this section for a comparative study. The problem is given below:

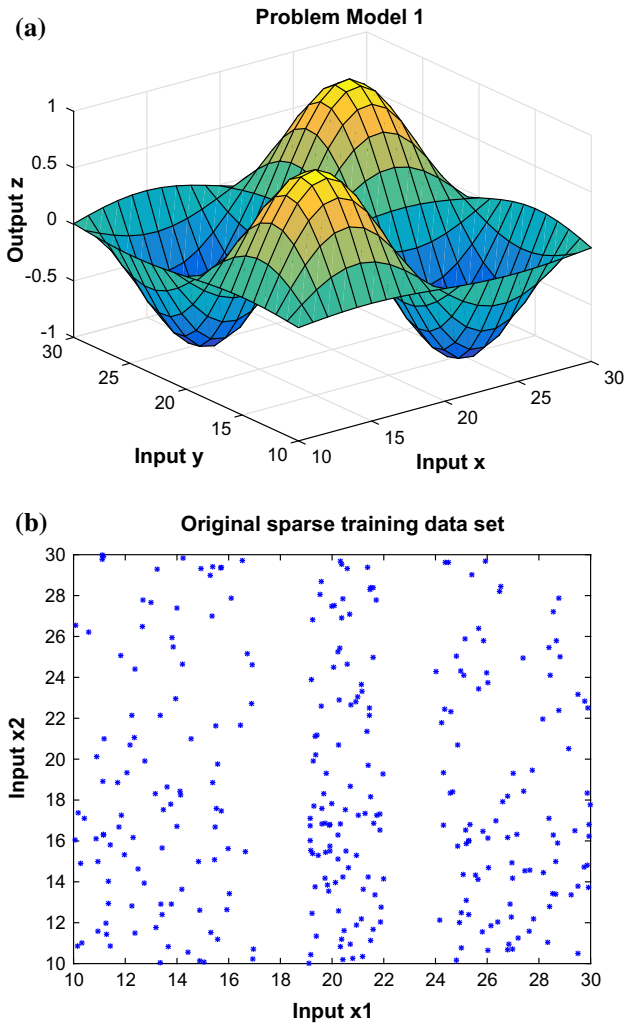

Fig. 5 Mathematical model in Experiment 1. **a** Surface view of Eq. 17. **b** Training data set distribution

$$f(x_1, x_2) = sin\left(\frac{x_1}{\pi}\right) \cdot sin\left(\frac{x_2}{\pi}\right), \tag{17}$$

which takes two inputs, x_1 ($x_1 \in [10, 30]$) and x_2 ($x_2 \in [10, 30]$), and produces a single output $y = f(x_1, x_2)$ ($y \in [-1, 1]$), as illustrated in Fig. 5a.

5.1.1 Rule base initialisation

In order to demonstrate the proposed TSK+ rule base generation approach, a sparse training data set $\mathbb{T}$ was manually generated from Eq. 17. The data set is composed of 300 data points sparsely distributed within the $[10, 30] \times [10, 30]$ domain covering 57% of the input domain. The distribution of this training data set is illustrated in a 2-dimensional plane in Fig. 5b. The key steps of TSK rule base initialisation using the training data set are summarised below.

Step 1 *Dense sub-data sets generation* The sparse training data set $\mathbb{T}$ was firstly divided into a number a dense sub-data

Table 1 SSE and performance improvement

No. of k	SSE	PI
1	2961.91	
2	2248.98	712.93
3	1763.74	485.24
4	1478.32	285.42
5	1286.41	191.91
6	1180.48	105.93
7	1079.48	101.00
8	997.626	81.85
9	946.238	51.39
10	880.43	65.81
11	840.865	39.57
12	808.51	32.36
13	767.61	40.90
14	709.997	57.61
15	689.301	20.70

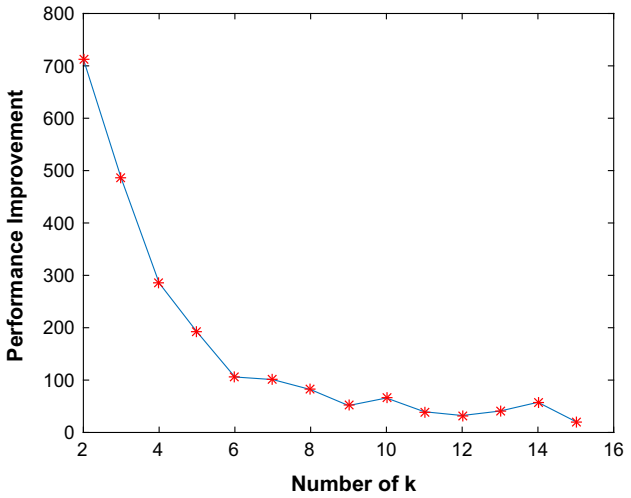

Fig. 6 Performance improvement against incremented k

sets by applying the sparse K-Means clustering algorithm. In particular, the number of sub-data sets was determined by the Elbow method as discussed in the Sect. 4.1.2. The performance improvements (PIs) against the incremented number clusters are listed in Table 1 and shown in Fig. 6. It is clear from the figure that the performance improvement decreases rapidly when k increased from 2 to 6, before it flattened out after 6. Therefore, 6 was taken as the number of clusters. The application of sparse K-Means led to 6 dense sub-data sets as demonstrated in Fig. 7.

Step 2 *Rule cluster generation* Once the sparse training data set $\mathbb{T}$ was divided into 6 dense sub-data sets ($T_1, T_2, \ldots, T_6$), the standard K-Means clustering algorithm was employed on each determined sub-data set to group similar data points into rule clusters. The application of the Elbow approach led

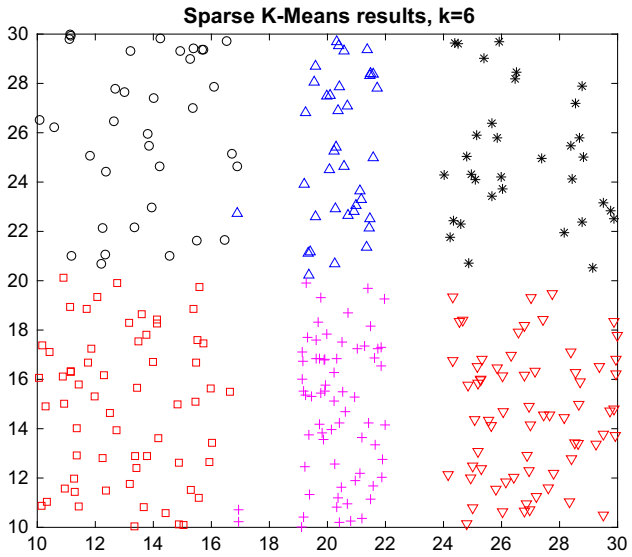

Fig. 7 Result of sparse K-Means where $k=6$

to a set of SSEs and PIs as listed in Table 2, which in turn determined the value of k for each sub-data set as listed in Table 3.

Step 3 *Rule base initialisation* A rule was extracted from each cluster and all the extracted rules jointly initialised the rule base. The generated 28 TSK fuzzy rules are detailed in Table 4.

5.1.2 TSK fuzzy interpolation

Given any input, overlapped with any rule antecedent or not, the proposed TSK+ inference engine is able to generate an output. For instance, a randomly generated testing data point was ($A_1^* = (27.37, 27.37, 27.37)$, $A_2^* = (13.56, 13.56, 13.56)$). The proposed TSK+ inference engine firstly calculated the similarity degrees between the given input and the antecedents of every rule ($S(A_1^*, A_{r1})$, $S(A_2^*, A_{r2})$, $r = \{1, 2, \ldots, 28\}$) using Eq. 7, with the results listed in Table 5.

From the calculated similarity degrees, the firing strength of each rule FS_r was computed using Eq. 2, and the sub-consequence from each rule was calculated using Eq. 4, which are also shown in Table 5. From this, the final output of the given input was calculated using Eq. 5 as $y = -0.902$.

5.1.3 Rule base optimisation

In order to achieve the optimal performance, the generated raw rule base was fine-tuned using the GA algorithm as detailed in Sect. 4.2. The GA parameters used in this experiment are listed in Table 6. The population was initialised as the individuals representing the raw rule base and its random variations. The performance against the number of iterations

Table 2 SSE and PI for each sub-data set ($k = \{1, 2, \ldots, 10\}$)

k	T_1		T_2		T_3		T_4		T_5		T_6	
	SSE	PI	SSE	PI	SSE	PI	SSE	PI	SSE	PI	SSE	PI
1	159.9		271.6		128.3		222.28		133.98		259.65	
2	109.42	50.48	187.65	83.95	68.61	59.69	140.16	82.12	96.06	37.92	187.95	71.70
3	81.44	27.98	149.88	37.78	56.66	11.95	114.23	25.93	66.42	29.64	147.02	40.93
4	66.85	14.86	126.35	23.53	46.16	10.50	99.48	14.75	52.46	13.96	123.73	23.29
5	56.45	10.13	109.73	16.62	39.87	6.29	85.06	14.42	42.79	9.67	109.39	14.34
6	48.04	8.41	94.14	15.59	33.04	6.83	77.22	7.84	37.69	5.10	95.25	14.104
7	42.01	6.03	83.67	10.47	28.33	4.71	69.90	7.32	33.52	4.17	85.26	9.99
8	36.22	5.79	77.63	6.04	26.26	2.07	61.87	8.03	29.18	4.34	80.06	5.2
9	32.4	3.82	74.63	8.00	24.03	2.23	56.17	5.70	25.01	4.17	74.62	5.44
10	28.77	3.63	67.11	7.52	22.69	1.34	52.39	3.78	24.29	0.72	68.56	6.06

Table 3 The value of k for each sub-data set

	T_1	T_2	T_3	T_4	T_5	T_6
Determined number of k	5	5	3	4	6	5

is shown in Fig. 8, which clearly demonstrates the performance improvements led by the GA.

5.1.4 Results comparison

In order to enable a direct comparative study with support of the approaches proposed in Bellaaj et al. (2013) and Tan et al. (2016), the proposed approach was also applied to 36 randomly generated testing data points. The sum of errors for the 36 testing data led by the proposed approach is shown in Table 7, in addition to those led by the compared approaches. The proposed TSK+ outperformed the approaches proposed in Bellaaj et al. (2013) and Tan et al. (2016), although less rules have been used. Another advantage of the proposed approach is that the number of rules led by the proposed system was determined automatically by the system without the requirement of any human inputs. In addition, noticeably, the optimisation process significantly improved the system performance by reducing the sum of error from 3.38 to 1.78.

5.2 Experiment 2

The proposed approach is not only able to deal with sparse or unevenly distributed data sets, but also able to work with dense data sets. In order to evaluate its ability in handling dense data sets, a two-input and single-output nonlinear mathematical model expressed in Eq. 18 was employed as a test bed, given that it has been used by Evsukoff et al. (2002) and Rezaee and Zarandi (2010).

$$f(x, y) = \frac{sin(x)}{x} \cdot \frac{sin(y)}{y}. \tag{18}$$

In this case, a randomly generated dense training data set was used, including 1681 data points distributed within the range of $[-10, 10] \times [-10, 10]$. By employing the proposed approach, 14 TSK fuzzy rules in total were generated. To allow a direct comparison, the mean-squares error (MSE) was used as the measurement of models, by following the work presented in Evsukoff et al. (2002) and zRezaee and Zarandi (2010). The detailed calculations of this experiment are omitted here. The MSE values led by different approaches with the specified number of rules are listed in Table 8. The results demonstrated that the proposed system TSK+ performed competitively with only 14 TSK fuzzy rules.

5.3 Experiment 3

This section considers a well-known real-world data set NSL-KDD-99 (Tavallaee et al. (2009)), which has been widely used as a benchmark (Bostani and Sheikhan (2017); Wang et al. (2010); Yang et al. (2017b)). This data set is a modified version of KDD Cup 99 data set generated in a military network environment. It contains 125,973 data instances with 41 attributes and 1 label which indicates the type of connection. In particular, normal connections and four types of attacks are labelled in the data set, including Denial of Service Attacks (DoS), User to Root Attacks(U2R), Remote to User Attacks (R2U) and Probes.

5.3.1 Data set pre-processing

The four most important attributes were selected from the original 41 using expertise knowledge in order to remove noises and redundancies in the work of Yang et al. (2017b). This experiment also took the four attributes as system input, and the application of automatic feature selection and reduc-

Table 4 Generated raw TSK rule base

No.	Input 1			Input 2			Output		
	a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	β_0	β_1	β_2
1	10.08	12.72	15.36	26.22	27.00	27.78	− 0.14	0.06	− 0.38
2	13.22	14.87	16.52	27.87	28.84	29.82	− 0.00	0.28	− 8.24
3	11.82	13.02	14.22	22.97	24.46	25.96	− 0.15	− 0.02	1.72
4	10.90	13.68	16.45	18.85	20.50	22.16	− 0.04	− 0.20	4.47
5	10.06	11.18	12.30	14.89	16.91	18.93	0.25	− 0.06	− 1.49
6	13.16	14.57	15.99	15.63	17.69	19.76	0.00	− 0.21	4.30
7	19.24	20.48	21.71	26.83	28.25	29.68	0.14	− 0.06	− 1.21
8	19.59	20.52	21.45	22.15	22.89	23.64	0.24	0.08	− 6.50
9	19.20	20.39	21.58	23.90	24.66	25.42	0.30	0.00	− 6.00
10	10.16	11.26	12.37	10.85	12.44	14.02	0.12	0.09	− 2.22
11	13.36	15.15	16.95	10.05	10.78	11.51	− 0.01	0.30	− 2.85
12	12.25	14.14	16.03	11.75	13.42	15.09	0.05	0.10	− 2.21
13	20.51	21.24	21.97	15.28	17.48	19.69	− 0.17	0.12	1.32
14	19.27	20.31	21.35	19.30	20.33	21.35	0.13	− 0.00	− 2.49
15	16.64	18.59	20.54	15.11	16.71	18.32	− 0.27	− 0.00	5.31
16	20.95	21.42	21.89	11.14	12.25	13.35	− 0.17	− 0.13	4.92
17	19.23	20.61	21.98	12.46	13.57	14.69	− 0.28	− 0.01	5.71
18	19.10	20.14	21.19	10.00	10.81	11.62	− 0.06	− 0.03	1.52
19	24.39	25.46	26.53	28.19	28.94	29.68	− 0.00	− 0.27	8.02
20	24.02	25.03	26.04	21.77	24.08	26.39	− 0.01	0.05	− 0.13
21	27.38	28.63	29.89	21.95	24.91	27.88	− 0.27	− 0.00	8.17
22	24.32	26.04	27.75	17.93	19.32	20.71	0.03	0.28	− 6.23
23	24.31	25.73	27.15	15.75	16.37	16.98	0.09	0.15	− 5.72
24	28.40	29.19	29.99	15.89	18.20	20.52	0.23	0.03	− 7.43
25	26.03	27.77	29.52	10.50	11.39	12.29	0.07	− 0.18	− 0.01
26	24.15	25.29	26.42	10.16	11.62	13.08	0.03	− 0.26	1.80
27	28.40	29.16	29.92	12.79	13.89	14.99	0.29	− 0.02	− 8.38
28	25.07	26.62	28.17	12.92	13.92	14.92	0.18	− 0.06	− 4.66

tion approaches, such as Zheng et al. (2015), remains as future work. The four selected input features are listed in Table 9. The data set in this experiment has also been normalised in an effort to reduce the potential noises in this real-world data set.

5.3.2 TSK+ model construction

As the labels are symbolic values, 0-order TSK-style fuzzy rules were used. In order to construct a 0-order TSK rule base, the training data set was divided into 5 sub-data sets based on the five symbolic labels, which are represented using five integer numbers. The sizes of the sub-data sets and their corresponding integer labels are listed in Table 10. The rule base generation process is summarised in four steps below.

Step 1 *Dense sub-data set generation* The sparse K-Means was applied on each sub-data set $\mathbb{T}_j$, $1 \leq j \leq 5$ to generate dense sub-data sets. Taking the second sub-data set $\mathbb{T}_2$ as an example, the performance improvement led by the increment of k_2, ($k_2 \in \{1, 2, \ldots, 10\}$) is shown in Fig. 9. Following the Elbow approach, 3 was selected as the number of clusters, i.e. $k_2 = 3$. Denote the 3 generated dense sub-sets as (T_{21}, T_{22}, T_{23}).

Step 2 *Rule cluster generation* The standard K-Means clustering algorithm was applied on T_{ji} ($j \in \{1, 2, \ldots, 5\}$ and i ranging from 1 to the determined cluster numbers using the Elbow approach), to generate rule clusters each representing a rule. Again, take $\mathbb{T}_2$ as an example for illustration. The determined cluster numbers k_{2i} for each dense sub-data set T_{2i} are shown in the third column of Table 11. Then, 13 rule clusters were generated from the sub-data set $\mathbb{T}_2$. The rule cluster generation process for other dense sub-data sets is not detailed here, but the generated rule cluster or the given training data set is summarised in Table 11.

Step 3 *Raw rule base generation* A rule is extracted from each generated rule cluster. Taking rule cluster RC_{12} as an

Table 5 Sub-result from each rule and its calculation details

p	$S(A_1^*, A_{p1})$	$S(A_2^*, A_{p2})$	FD_p	Consequence
1	0.0053	0.059	0.053	− 0.014
2	0.020	0.003	0.003	− 0.012
3	0.004	0.010	0.004	− 0.014
4	0.001	0.836	0.001	0.006
5	0.007	0.455	0.007	0.004
6	0.016	0.780	0.016	0.023
7	0.467	0.157	0.157	0.170
8	0.461	0.004	0.004	0.008
9	0.449	0.052	0.052	0.118
10	0.012	0.954	0.012	0.018
11	0.024	0.870	0.024	0.024
12	0.002	0.939	0.002	0.005
13	0.211	0.848	0.211	− 0.430
14	0.567	0.812	0.567	− 0.996
15	0.440	0.480	0.440	0.114
16	0.591	0.941	0.591	− 0.913
17	0.480	0.969	0.480	− 1.042
18	0.414	0.871	0.414	− 0.232
19	0.926	0.009	0.009	0.005
20	0.932	0.004	0.004	0.014
21	0.925	0.077	0.077	0.005
22	0.927	0.868	0.868	− 0.937
23	0.918	0.736	0.736	− 0.471
24	0.932	0.616	0.616	− 1.030
25	0.948	0.902	0.902	− 0.622
26	0.947	0.966	0.947	− 0.547
27	0.906	0.913	0.906	− 0.849
28	0.920	0.964	0.920	− 0.589

Table 6 GA parameters

Parameters	Values
Population Size	100
Max value in fitness function	2
Crossover rate	0.85
Mutation rate	0.05
Maximum iteration	10,000
Termination threshold	0.01

example, which is the first determined rule cluster in T_{21}, the corresponding rule R_{12} can be extracted. In particular, the consequence of rule R_{12} is the integer number representing the class of connections; and the rule antecedents are four triangle fuzzy sets ($A_{(12)1}$, $A_{(12)2}$, $A_{(12)3}$, $A_{(12)4}$) led by the approach discussed in Sect. 4.1.4. The generated rule R_{12} is:

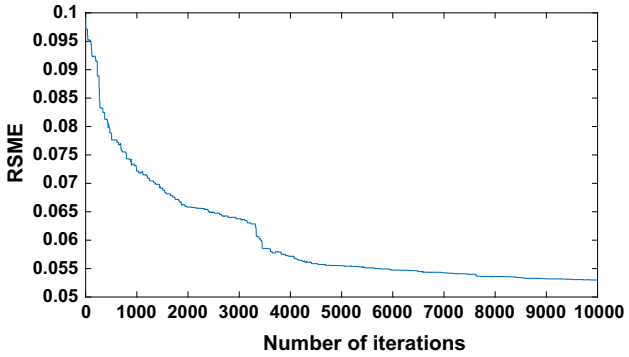

Fig. 8 RMSE values decrease over time during optimisation

Table 7 Results led by approaches in experiment 1

Approaches	No. of rules	Sum of error	RSME
Bellaaj et al. (2013)	36	5.8	0.161
Tan et al. (2016)	36	3.1	0.086
TSK+ without GA	28	3.38	0.094
TSK+	28	1.78	**0.067**

Bold value indicates best performance

Table 8 Results led by fuzzy models in experiment 2

Approaches	No. of rule	MSE
Evsukoff et al. (2002)	49	0.001
Rezaee and Zarandi (2010)	13	0.0004
Rezaee and Zarandi (2010)	14	0.0002
TSK+	14	0.0002

Table 9 Selected input features

Feature #	Feature
5	Size from source to destination
6	Size from destination to source
23	Number of connections in past 2 second
35	Different services rate for destination host

$$R_{12}: \textbf{IF } x_1 \text{ is } (0.588, 0.599, 0.601) \text{ and } x_2 \text{ is } (0, 0.387, 0.414) \text{ and } x_3 \text{ is } (0.05, 0.075, 0.1) \text{ and } x_4 \text{ is } (0, 0, 0) \textbf{ THEN } y = 2. \quad (19)$$

According to Table 11, 46 rules in total were generated to initialise the rule base. The detailed initialised rule base can be found in Appendix I.

Step 4 *Rule base optimisation* The GA was applied to optimise the membership functions of the fuzzy sets involved in the extracted fuzzy rules. The same GA parameters used in

Table 10 Data details regarding the types of connections

Datasets	No. of instances	Classes	Rule consequence
$\mathbb{T}_1$	53,874	Normal traffic	1
$\mathbb{T}_2$	36,741	DoS	2
$\mathbb{T}_3$	42	U2R	3
$\mathbb{T}_4$	796	R2U	4
$\mathbb{T}_5$	9,325	Probes	5

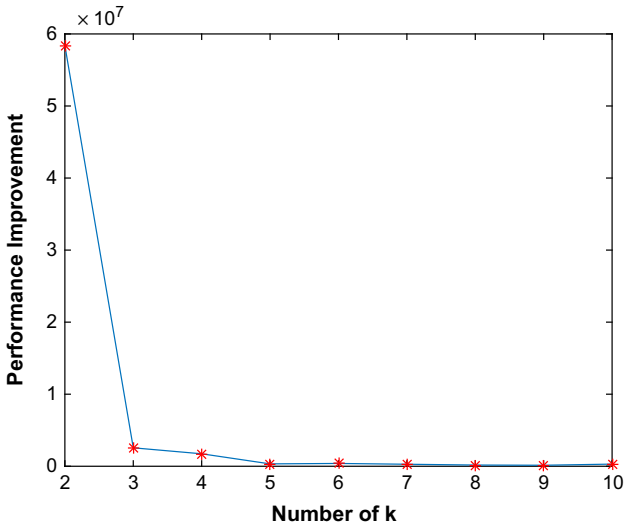

Fig. 9 Performance improvement regarding incremented k

Experiment 1 as listed in Table 6 were also used in this example, but the number of iterations was increased to 20,000. The system performance against the number of iterations used in GA is shown in Fig. 10. The optimised rule base is attached in Appendix II.

5.3.3 TSK+ model evaluation

Once the TSK fuzzy rule base was generated, it can then be used for connection classification by the proposed TSK+ inference engine. The rule base and the inference engine TSK+ jointly formed the fuzzy model, which was validated and evaluated using a testing dataset. The testing data set contains 22,544 data samples provided by Tavallaee et al. (2009). The testing data set was also extracted from original

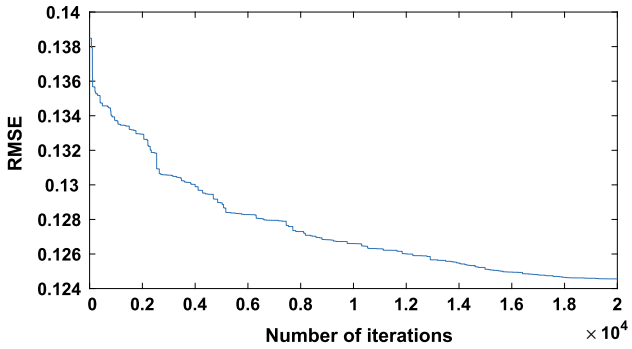

Fig. 10 RMSE decreasing over time

KDD Cup 99 data set, but it does not share any data instance with the training data set NSL-KDD-99.

Note that the testing data set has been used in a number of projects with different classification approaches. In particular, decision tree, naive Bayes, back-propagation neural network (BPNN), fuzzy clustering-artificial neural network (FC-ANN) have been employed in Wang et al. (2010), and modified optimum-path forest (MOPF) was applied in Bostani and Sheikhan (2017). The accuracy of the classification results for each class of network traffic generated by different approaches including the proposed one with the initialised rule base and the optimised rule base is listed in Table 12.

Table 12 Performance comparison

	Normal Traffic	DoS	U2R	R2U	Probes
Decision tree Wang et al. (2010)	91.22	97.24	15.38	1.43	78.13
Naive bayes Wang et al. (2010)	89.22	96.65	7.69	8.57	88.13
BPNN Wang et al. (2010)	89.75	97.20	23.08	5.71	**88.75**
FC-ANN Wang et al. (2010)	91.32	96.70	76.92	58.57	80.00
MOPF Bostani and Sheikhan (2017)	N/A	96.89	**77.98**	81.13	85.92
TSK+ without GA (the proposed approach)	77.10	94.07	57.69	55.29	78.71
TSK+ (the proposed approach)	**93.10**	**97.84**	65.38	**84.65**	85.69

Bold values indicate best performance

Table 11 Rules and clusters for each set of data instances with the same type of connection

Dense data set	Normal $\mathbb{T}_1$			DoS $\mathbb{T}_2$			R2U $\mathbb{T}_3$	U2R $\mathbb{T}_4$			Probes $\mathbb{T}_5$		
	T_{11}	T_{12}	T_{13}	T_{21}	T_{22}	T_{23}	T_{31}	T_{41}	T_{42}	T_{43}	T_{51}	T_{52}	T_{53}
Index i of rule cluster RC_i	1–4	5–7	8–11	12–14	15–18	19–24	25–30	31–35	36–38	39–41	42–44	45	46
Index i of rule R_i	1–4	5–7	8–11	12–14	15–18	19–24	25–30	31–35	36–38	39–41	42–44	45	46

The results show that the proposed TSK+ overall outperformed all other approaches; and the proposed rule base optimisation approach significantly improved the system performance by over 10% on average. In particular, the proposed system achieved better accuracies on the prediction of normal connections, DoS and R2U than those of all other approaches; worse performance led by the proposed approach in the class of U2R compared to FC-ANN and MOPF; and similar result was generated for class Probes with the existing best performance resulted from other approaches.

6 Conclusion

This paper proposed a fuzzy inference system TSK+, which extended the conventional TSK inference system such that it is also applicable to sparse rule bases and unevenly distributed rule bases. This is achieved by allowing the interpolation and extrapolation of the output from all rules even if the given input does not overlap with any rule antecedents. This paper also proposed a novel data-driven rule base generation approach, which is workable with spare data sets, dense data set, and unevenly distributed data sets. The system was validated and evaluated by applying two benchmark problems and one real-world data set. The experimental results demonstrated the wide applicability of the proposed system with compact rule bases and competitive performances.

The proposed system can be further enhanced in the following areas. Firstly, the sparsity-aware possibilistic clustering algorithm (Xenaki et al. (2016)) was designed to work with sparse data sets, and thus, it is desired to investigate how this clustering algorithm can support the proposed TSK+ inference system. Secondly, it is worthwhile to study how the proposed sparse rule base generation approach can help in generating Mamdani-style fuzzy rule bases. Finally, it would be interesting to define the sparsity or density of a data set such that more accurate clustering results can be generated during rule cluster generation.

Acknowledgements This work is supported by a Northumbria University PhD scholarship, the National Natural Science Foundation of China (No. 61502068) and the China Postdoctoral Science Foundation (Nos. 2013M541213 and 2015T80239).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Appendix I

See Table 13

Table 13 Initialised Rule Base

i	Input												Output
	A_{i1}			A_{i2}			A_{i3}			A_{i4}			f_i
	$a_{(i1)1}$	$a_{(i1)2}$	$a_{(i1)3}$	$a_{(i2)1}$	$a_{(i2)2}$	$a_{(i2)3}$	$a_{(i3)1}$	$a_{(i3)2}$	$a_{(i3)3}$	$a_{(i4)1}$	$a_{(i4)2}$	$a_{(i4)3}$	
1	0.000	0.210	0.714	0.000	0.228	0.438	0.000	0.100	1.000	0.000	0.000	1.000	1
2	0.000	0.212	0.711	0.000	0.520	0.787	0.025	0.100	0.599	0.000	0.000	1.000	1
3	0.000	0.233	0.450	0.805	0.833	0.834	0.025	0.025	0.125	0.000	0.020	0.430	1
4	0.723	0.760	0.843	0.000	0.000	0.723	0.025	0.050	0.050	0.000	0.040	1.000	1
5	0.833	0.833	0.833	0.000	0.000	0.788	0.025	0.050	0.150	0.000	0.040	1.000	1
6	0.834	0.834	0.834	0.689	0.830	0.833	0.025	0.025	0.050	0.040	0.185	0.360	1
7	0.834	0.834	0.834	0.000	0.670	0.761	0.025	0.025	0.075	0.000	0.090	0.610	1
8	0.836	0.836	0.836	0.561	0.833	0.833	0.025	0.025	0.025	0.030	0.055	0.080	1
9	0.848	0.848	0.848	0.834	0.834	0.834	0.025	0.025	0.025	0.030	0.030	0.030	1
10	0.837	0.837	0.837	0.682	0.687	0.689	0.025	0.025	0.025	0.030	0.030	0.030	1
11	0.835	0.835	0.835	0.000	0.747	0.783	0.025	0.050	0.050	0.090	0.160	0.920	1
12	0.588	0.599	0.601	0.000	0.387	0.414	0.050	0.075	0.100	0.000	0.000	0.000	2
13	0.000	0.000	0.263	0.000	0.000	0.047	0.554	0.584	0.623	0.000	0.070	1.000	2
14	0.000	0.000	0.272	0.000	0.000	0.000	0.025	0.524	0.554	0.000	0.050	0.760	2
15	0.593	0.601	0.601	0.468	0.487	0.487	0.025	0.075	0.725	0.000	0.000	0.000	2

Table 13 continued

i	Input												Output
	A_{i1}			A_{i2}			A_{i3}			A_{i4}			f_i
	$a_{(i1)1}$	$a_{(i1)2}$	$a_{(i1)3}$	$a_{(i2)1}$	$a_{(i2)2}$	$a_{(i2)3}$	$a_{(i3)1}$	$a_{(i3)2}$	$a_{(i3)3}$	$a_{(i4)1}$	$a_{(i4)2}$	$a_{(i4)3}$	
16	0.000	0.000	0.263	0.000	0.000	0.000	0.624	0.663	0.710	0.000	0.070	0.750	2
17	0.580	0.585	0.587	0.414	0.468	0.487	0.050	0.100	0.150	0.000	0.000	0.000	2
18	0.554	0.568	0.579	0.360	0.401	0.487	0.025	0.088	0.150	0.000	0.000	0.000	2
19	0.352	0.352	0.352	0.000	0.000	0.000	0.025	0.543	0.605	0.000	0.020	0.260	2
20	0.352	0.352	0.352	0.000	0.000	0.000	0.752	1.000	1.000	0.000	0.020	0.260	2
21	0.360	0.360	0.520	0.000	0.000	0.330	0.025	0.050	0.659	0.000	0.020	1.000	2
22	0.352	0.352	0.352	0.000	0.000	0.000	0.606	0.667	0.751	0.000	0.000	0.260	2
23	0.263	0.263	0.263	0.000	0.000	0.000	0.711	0.970	0.993	0.000	0.000	0.000	2
24	0.524	0.535	0.549	0.360	0.487	0.487	0.050	0.100	0.150	0.000	0.000	0.000	2
25	0.000	0.000	0.362	0.406	0.439	0.498	0.025	0.025	0.075	0.000	0.000	0.020	3
26	0.000	0.149	0.219	0.000	0.296	0.371	0.025	0.025	0.100	0.000	0.000	0.750	3
27	0.215	0.362	0.365	0.368	0.384	0.390	0.025	0.025	0.599	0.000	0.000	0.060	3
28	0.376	0.378	0.382	0.404	0.405	0.417	0.025	0.025	0.050	0.000	0.000	0.000	3
29	0.244	0.244	0.244	0.631	0.631	0.631	0.025	0.025	0.025	0.000	0.000	0.000	3
30	0.365	0.407	0.449	0.531	0.538	0.545	0.025	0.025	0.025	0.000	0.010	0.020	3
31	0.000	0.229	0.358	0.000	0.000	0.362	0.025	0.025	0.100	0.000	0.000	0.670	4
32	0.000	0.356	0.359	0.371	0.378	0.573	0.025	0.025	0.050	0.000	0.020	1.000	4
33	0.000	0.305	0.361	0.754	0.771	0.787	0.025	0.025	0.025	0.000	0.000	0.000	4
34	0.000	0.000	0.000	0.850	0.833	0.833	0.025	0.025	0.025	0.000	0.000	0.000	4
35	0.464	0.464	0.464	0.000	0.000	0.000	0.025	0.050	0.075	0.000	0.000	0.220	4
36	0.834	0.834	0.834	0.000	0.000	0.000	0.025	0.025	0.025	0.000	0.000	0.290	4
37	0.834	0.834	0.834	0.000	0.000	0.000	0.025	0.025	0.025	0.000	0.000	0.290	4
38	0.834	0.834	0.834	0.000	0.000	0.000	0.025	0.025	0.050	0.000	0.000	0.130	4
39	0.000	0.000	0.000	0.834	0.834	0.834	0.025	0.025	0.025	0.000	0.000	0.000	4
40	0.000	0.000	0.000	0.834	0.834	0.834	0.025	0.050	0.075	0.000	0.000	0.000	4
41	0.000	0.000	0.000	0.834	0.834	0.834	0.025	0.025	0.025	0.000	0.000	0.000	4
42	0.000	0.002	0.836	0.000	0.000	0.596	0.000	0.025	1.000	0.000	0.270	1.000	5
43	0.897	0.897	0.897	0.000	0.000	0.000	0.025	0.025	0.025	0.120	0.120	0.120	5
44	0.869	0.869	0.869	0.000	0.000	0.000	0.025	0.025	0.025	0.110	0.110	0.110	5
45	0.000	0.000	0.000	0.900	0.900	1.000	0.025	0.025	0.050	0.500	0.500	0.650	5
46	0.937	0.988	1.000	0.000	0.000	0.000	0.025	0.025	0.538	0.080	0.105	0.120	5

Appendix II

See Table 14

Table 14 Optimised Rule Base

i	Input												Output
	A_{i1}			A_{i2}			A_{i3}			A_{i4}			f_i
	$a_{(i1)1}$	$a_{(i1)2}$	$a_{(i1)3}$	$a_{(i2)1}$	$a_{(i2)2}$	$a_{(i2)3}$	$a_{(i3)1}$	$a_{(i3)2}$	$a_{(i3)3}$	$a_{(i4)1}$	$a_{(i4)2}$	$a_{(i4)3}$	
1	0.000	0.199	0.332	0.000	0.210	0.350	0.000	0.157	0.261	0.000	0.157	0.262	1
2	0.000	0.138	0.184	0.000	0.124	0.166	0.000	0.178	0.226	0.237	0.675	0.868	1
3	0.000	0.167	0.331	0.000	0.176	0.333	0.746	0.782	0.774	0.175	0.480	0.750	1
4	0.333	0.453	0.667	0.317	0.455	0.700	0.000	0.081	0.225	0.000	0.109	0.289	1
5	0.348	0.567	0.731	0.000	0.000	0.000	0.000	0.116	0.204	0.000	0.086	0.150	1
6	0.475	0.787	0.829	0.301	0.599	0.602	0.000	0.235	0.237	0.000	0.260	0.276	1
7	0.602	0.873	0.995	0.300	0.529	0.632	0.000	0.164	0.226	0.000	0.173	0.250	1
8	0.750	0.780	0.950	0.000	0.026	0.175	0.000	0.071	0.474	0.000	0.036	0.238	1
9	0.831	0.912	1.000	0.873	0.932	0.995	0.000	0.108	0.214	0.000	0.120	0.249	1
10	0.833	0.889	1.000	0.000	0.013	0.158	0.000	0.024	0.304	0.000	0.021	0.262	1
11	0.916	0.936	0.998	0.317	0.503	0.632	0.000	0.154	0.249	0.000	0.148	0.250	1
12	0.000	0.152	0.175	0.000	0.118	0.135	0.748	0.920	0.945	0.000	0.252	0.289	2
13	0.572	0.701	0.774	0.000	0.152	0.317	0.788	0.866	0.998	0.000	0.132	0.276	2
14	0.632	0.896	0.950	0.000	0.305	0.368	0.000	0.197	0.249	0.000	0.218	0.263	2
15	0.500	0.717	0.752	0.000	0.259	0.317	0.746	1.000	1.000	0.000	0.203	0.236	2
16	0.745	0.753	0.775	0.000	0.307	0.349	0.080	0.081	0.072	0.000	0.000	0.000	2
17	0.461	0.515	0.691	0.275	0.334	0.404	0.040	0.119	0.145	0.000	0.000	0.000	2
18	0.323	0.498	0.695	0.432	0.498	0.579	0.018	0.086	0.155	0.000	0.000	0.000	2
19	0.281	0.324	0.380	0.000	0.000	0.000	0.026	0.586	0.680	0.000	0.018	0.459	2
20	0.000	0.049	0.158	0.000	0.057	0.184	0.474	0.637	1.000	0.000	0.078	0.250	2
21	0.298	0.377	0.427	0.000	0.000	0.320	0.027	0.038	0.671	0.000	0.009	0.760	2
22	0.177	0.228	0.442	0.000	0.000	0.000	0.577	0.631	0.648	0.000	0.000	0.168	2
23	0.244	0.270	0.326	0.000	0.000	0.000	0.665	0.765	0.836	0.000	0.000	0.000	2
24	0.792	0.848	0.948	0.000	0.000	0.735	0.000	0.090	0.249	0.000	0.095	0.263	2
25	0.000	0.000	0.303	0.365	0.394	0.647	0.021	0.022	0.073	0.000	0.000	0.025	3
26	0.551	0.831	0.919	0.602	0.904	1.000	0.000	0.200	0.263	0.000	0.181	0.238	3
27	0.543	0.815	0.842	0.831	1.000	1.000	0.000	0.214	0.226	0.000	0.249	0.263	3
28	0.000	0.157	0.333	0.498	0.667	0.831	0.000	0.102	0.238	0.500	0.745	1.000	3
29	0.150	0.263	0.276	0.525	0.641	0.705	0.023	0.026	0.033	0.000	0.000	0.000	3
30	0.320	0.437	0.461	0.477	0.498	0.758	0.022	0.027	0.033	0.000	0.014	0.022	3
31	0.000	0.144	0.150	0.833	1.000	1.000	0.000	0.072	0.249	0.000	0.076	0.263	4
32	0.000	0.357	0.466	0.296	0.428	0.838	0.026	0.028	0.036	0.000	0.022	0.755	4
33	0.000	0.154	0.367	0.633	0.746	0.903	0.000	0.100	0.226	0.000	0.099	0.236	4
34	0.000	0.019	0.175	0.000	0.031	0.286	0.000	0.028	0.250	0.000	0.029	0.262	4
35	0.300	0.428	0.528	0.000	0.000	0.000	0.035	0.042	0.085	0.000	0.000	0.249	4
36	0.831	1.000	1.000	0.000	0.096	0.150	0.000	0.152	0.237	0.000	0.160	0.238	4
37	0.386	0.477	0.663	0.000	0.047	0.150	0.000	0.082	0.249	0.000	0.078	0.238	4
38	0.665	0.846	1.000	0.664	0.930	1.000	0.000	0.141	0.262	0.000	0.134	0.248	4
39	0.000	0.144	0.150	0.833	1.000	1.000	0.000	0.072	0.249	0.000	0.076	0.263	4
40	0.000	0.000	0.000	0.663	0.661	1.000	0.032	0.036	0.040	0.000	0.000	0.000	4

Table 14 continued

i	Input												Output
	A_{i1}			A_{i2}			A_{i3}			A_{i4}			f_i
	$a_{(i1)1}$	$a_{(i1)2}$	$a_{(i1)3}$	$a_{(i2)1}$	$a_{(i2)2}$	$a_{(i2)3}$	$a_{(i3)1}$	$a_{(i3)2}$	$a_{(i3)3}$	$a_{(i4)1}$	$a_{(i4)2}$	$a_{(i4)3}$	
41	0.000	0.000	0.000	0.671	0.701	0.807	0.030	0.023	0.030	0.000	0.000	0.000	4
42	0.000	0.000	0.175	0.000	0.000	0.166	0.000	0.000	0.250	0.750	0.750	0.815	5
43	0.000	0.160	0.166	0.000	0.152	0.158	0.000	0.228	0.238	0.525	0.930	0.948	5
44	0.000	0.023	0.166	0.000	0.026	0.175	0.750	0.765	0.857	0.522	0.606	0.950	5
45	0.831	0.857	0.857	0.000	0.146	0.150	0.000	0.219	0.238	0.000	0.458	0.473	5
46	0.000	0.003	0.158	0.875	0.881	0.987	0.000	0.006	0.275	0.275	0.285	0.788	5

References

Baker JE (1985) Adaptive selection methods for genetic algorithms. In: Proceedings of an international conference on genetic algorithms and their applications, Hillsdale, New Jersey, pp 101–111

Bellaaj H, Ketata R, Chtourou M (2013) A new method for fuzzy rule base reduction. J Intell Fuzzy Syst. 25(3):605–613

Bostani H, Sheikhan M (2017) Modification of supervised opf-based intrusion detection systems using unsupervised learning and social network concept. Pattern Recognit 62:56–72

Chang PC, Fan CY (2008) A hybrid system integrating a wavelet and tsk fuzzy rules for stock price forecasting. IEEE Trans Syst Man, and Cybern Part C Appl Rev 38(6):802–815

Chen MY, Linkens D (2004) Rule-base self-generation and simplification for data-driven fuzzy models. Fuzzy Sets and Syst 142:243–265

Chen S, Hsin W (2015) Weighted fuzzy interpolative reasoning based on the slopes of fuzzy sets and particle swarm optimization techniques. IEEE Trans Cybern 45(7):1250–1261

Chen SJ, Chen SM (2003) Fuzzy risk analysis based on similarity measures of generalized fuzzy numbers. IEEE Trans Fuzzy Syst 11(1):45–56

Evsukoff A, Branco AC, Galichet S (2002) Structure identification and parameter optimization for non-linear fuzzy modeling. Fuzzy Sets and Syst 132(2):173–188

Huang Z, Shen Q (2006) Fuzzy interpolative reasoning via scale and move transformations. IEEE Trans Fuzzy Syst 14(2):340–359

Huang Z, Shen Q (2008) Fuzzy interpolation and extrapolation: A practical approach. IEEE Trans Fuzzy Syst 16(1):13–28

Johanyák ZC, Kovács S (2005) Distance based similarity measures of fuzzy sets. In: Proceedings of SAMI 2005

Kerk YW, Pang LM, Tay KM, Lim CP (2016) Multi-expert decision-making with incomplete and noisy fuzzy rules and the monotone test. In: 2016 IEEE international conference on fuzzy systems (FUZZ-IEEE), pp 94–101

Kóczy L, Hirota K (1993) Approximate reasoning by linear rule interpolation and general approximation. Int J Approx Reason 9(3):197–225

Kóczy L, Hirota K (1997) Size reduction by interpolation in fuzzy rule bases. IEEE Trans Syst Man Cybern Part B Cybern 27(1):14–25

Kodinariya TM, Makwana PR (2013) Review on determining number of cluster in k-means clustering. Int J 1(6):90–95

Li J, Qu Y, Shum HPH, Yang L (2017) TSK inference with sparse rule bases. Springer International Publishing, Cham, pp 107–123

MacQueen J (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Oakland, CA, USA., pp 281–297

Mamdani EH (1977) Application of fuzzy logic to approximate reasoning using linguistic synthesis. Comput IEEE Trans C 26(12):1182–1191

Mucientes M, Vidal JC, Bugarín A, Lama M (2009) Processing time estimations by variable structure tsk rules learned through genetic programming. Soft Comput 13(5):497–509

Naik N, Diao R, Shen Q (2014) Genetic algorithm-aided dynamic fuzzy rule interpolation. In: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) 2014, pp 2198–2205

Nandi AK, Klawonn F (2007) Detecting ambiguities in regression problems using tsk models. Soft Comput 11(5):467–478

Negnevitsky M (2005) Artificial intelligence: a guide to intelligent systems. Pearson Education, London

Rezaee B, Zarandi MF (2010) Data-driven fuzzy modeling for Takagi-Sugeno-Kang fuzzy system. Inf Sci 180(2):241–255

Shen Q, Yang L (2011) Generalisation of scale and move transformation-based fuzzy interpolation. JACIII 15(3):288–298. https://doi.org/10.20965/jaciii.2011.p0288

Takagi T, Sugeno M (1985) Fuzzy identification of systems and its applications to modeling and control. Syst Man and Cybern IEEE Trans SMC 15(1):116–132

Tan Y, Li J, Wonders M, Chao F, Shum HPH, Yang L (2016) Towards sparse rule base generation for fuzzy rule interpolation. In: IEEE world congress on computation intelligence internation conference

Tavallaee M, Bagheri E, Lu W, Ghorbani A (2009) A detailed analysis of the kdd cup 99 data set. In: The second IEEE symposium on computational intelligence for security and defence applications

Wang G, Hao J, Ma J, Huang L (2010) A new approach to intrusion detection using artificial neural networks and fuzzy clustering. Exp Syst Appl 37(9):6225–6232

Witten DM, Tibshirani R (2010) A framework for feature selection in clustering. J Am Stat Assoc 105(490):713–726

Xenaki SD, Koutroumbas KD, Rontogiannis AA (2016) Sparsity-aware possibilistic clustering algorithms. IEEE Trans Fuzzy Syst 24(6):1611–1626

Yang L, Shen Q (2011) Adaptive fuzzy interpolation. Fuzzy Syst IEEE Trans 19(6):1107–1126

Yang L, Shen Q (2013) Closed form fuzzy interpolation. Fuzzy Sets and Syst 225:1–22 theme: Fuzzy Systems

Yang L, Chao F, Shen Q (2017) Generalized adaptive fuzzy rule interpolation. IEEE Trans Fuzzy Syst 25(4):839–853

Yang L, Li J, Fehringer G, Barraclough P, Sexton G, Cao Y (2017) Intrusion detection system by fuzzy interpolation. In: 2017 IEEE international conference on fuzzy systems

Yang L, Zuo Z, Chao F, Qu Y (2017) Fuzzy rule interpolation and its application in control. In: Ramakrishnan S (ed) Modern Fuzzy Control Systems and Its Applications, InTech, Rijeka

Zheng L, Diao R, Shen Q (2015) Self-adjusting harmony search-based feature selection. Soft Comput 19(6):1567–1579