LUPINE PUBLISHERS

Current Trends in Computer Sciences & Applications

DOI: 10.32474/CTCSA.2019.01.000115

ISSN: 2643-6744

Research Article

Complex Neural Fuzzy Prediction Using Multi-Swarm Continuous Ant Colony Optimization

Chunshien Li[1]* and Wei-Chu Weng[2]

[1]*Department of Information Management, Nation Central University, Taiwan*

[2]*Department of Systems Engineering and Naval Architecture, National Taiwan Ocean University, Taiwan*

***Corresponding author:** Chunshien Li, Department of Information Management, Nation Central University No. 300, Zhongli Taoyuan City 32001, Taiwan

Received: June 14, 2019 **Published:** July 02, 2019

Abstract

Prediction of time series is one of major research subjects in data science. This paper proposes a novel approach to the problem of multiple-target prediction. The proposed approach is mainly composed of three parts: the complex neuro-fuzzy system (CNFS) built by using complex fuzzy sets, the two-stage feature selection method for multiple targets, and the hybrid machine learning method that uses the multi-swarm continuous ant colony optimization (MCACO) and the recursive least squares estimation (RLSE). The CNFS predictive model is responsible for prediction after training. During the training of the model, the parameters are updated by the MCACO method and the RLSE method where the two methods work cooperatively to become one machine learning procedure. For the predictive model, complex fuzzy sets (CFSs) are with complex-valued membership degrees within the unit disk of the complex plane, useful to the non-linear mapping ability of the CNFS model for multiple target prediction. This CFS property is contrast to real-valued membership degrees in the unit interval [0,1] of traditional fuzzy sets. The two-stage feature selection applies to select significant features to be the inputs to the model for multiple target prediction. Experiments using real world data sets obtained from stock markets for the prediction of multiple targets have been conducted. With the results and performance comparison, the proposed approach has shown outstanding performance over other compared methods.

Keywords: Feature selection for multiple targets; Complex fuzzy set; Continuous ant colony optimization; Multiple swarms; Multiple target prediction

Introduction

Prediction of time series is one of major research subjects in data science. When data grows massively, effective prediction becomes much important. A novel approach is therefore proposed to effectively make multiple-targets prediction. It uses a complex neuro-fuzzy system where the involved parameters are decided by the proposed method MCACO-RLSE, integrating both the multi-swarm continuous ant colony optimization (MCACO) algorithm and the well-known recursive least squares estimation (RLSE) method. In addition, a prediction system is developed to demonstrate the performance. The prediction system adopts a neuro-fuzzy system composed of several fuzzy if-then rules imbedded in neural network structure, where complex fuzzy sets and Takagi-Sugeno linear functions are used to define the premises and the consequents of rules, respectively. The adopted system is termed as Takagi-Sugeno complex neuro-fuzzy system (denoted as TS-CNFS in short). All parameters in the if-part are optimized by the method MCACO and those in the *then*-part are estimated by the method of RLSE. The method MCACO is a kind of continuous ant colony optimization (CACO) where multiple ant colonies are used. Furthermore, a two-stage feature selection is applied to select the input data of the most-affecting to multiple-targets. The method MCACORLSE associates two methods to decide all involved parameters separately and cooperatively. It comes out that the MCACO-RLSE helps the optimization effectively in the massively deflated solution space. The computation time is also decreased significantly. In addition, the two-stage feature selection presented in this paper selects only a few inputs to make multiple-targets prediction.

In the paper, practical stock prices and indices are chosen as case studies. Stock market forecasting is an important investment issue. Early in 1992, Engle [1] proposed ARCH model that successfully estimated the variance of economic inflation. Moreover, in 1986, Bollerslev [2] proposed the GARCH model that generalized the ARCH model. There are many implicit and complex factors making stock market forecasting difficult. Thus, technologies of machine learning were introduced. In 1943, McCulloch and Pitts proposed that a neuron works like a switch connecting with other neurons when turned on and vice versa. In 1959, Widrow & Haff [3] developed a neuro network model of self-adapting linear unit (ADALINE). After having been trained, this model is capable to solve practical problems such as weather forecasting. In 1990, Kimoto et al. [4] applied neural networks to stock market prediction. In 2000, Kim & Han [5] predicted stock market using a neural system optimized by genetic algorithms. In 2007, Roh [6] applied neural networks to forecasting the volatility of stock price index. Stock market forecasting is a problem with real values, although initially simple neural networks deal with binary data. In 1965, Zadeh [7] proposed the concept of fuzzy sets, converting two-valued membership in {0,1} to continuous membership in [0,1] through membership function. Furthermore, in 2002, Romat [8] proposed an advanced concept of complex fuzzy sets, expanding membership from a continuous segment [0,1] to a two-dimensional complex plane of unit disk. The difference between them is on the phase dimension of complex-valued membership initiated by complex fuzzy sets. The adaptive TS-CNFS is based on Takagi-Sugeno fuzzy system that Takagi & Sugeno [9] firstly proposed in 1985. In 2013, Li & Chiang [10] developed an advanced CNFS system with ARIMA (denoted as CNFS-ARIMA) that successfully implemented dual-output forecasting and performed excellent. This CNFS-ARIMA system adopted Gaussian functions and their derivatives to implement the complex-valued membership functions of complex fuzzy sets, which are therefore used in the paper. The method of MCACO is an algorithm of multiple ant colonies using continuous ant colony optimization (CACO). In 1991, Dorigo [11] proposed ant colony optimization (ACO) for optimizing traveling salesmen problem whose solution space is discrete. To deal with problems in a continuous solution space, Dorigo [12] further developed the method of CACO in 2008. The CACO method uses probability density to demonstrate how ants select their own routes in a continuous space for searching food. This paper presents the MCACO algorithm hybrid with the RLSE method to boost the machine learning ability of searching solution for the proposed TS-CNFS used on forecasting stock market. Data collected in the real world may contain redundant or noisy annal recordings that easily cause false prediction. Feature selection is a way to select the most-affecting data as the input for effective prediction. There are several methods for feature selection. The paper presents the two-stage feature selection using the theory of Shannon information entropy. In 1949, Shannon [13] proposed a theory of information entropy that quantified the level of information uncertainty. When data are real values, probability density, instead of probability, is used. In the paper, the method of kernel density estimation is adopted for evaluating probability density [14-16].

The rest contents of the paper are organized as follows. Section 2 describes the proposed TS-CNFS predictive system and the novel MCACO-RLSE method for machine learning. Section 3 describes the proposed two-stage feature selection crossing multiple targets. Section 4 demonstrates the experimentation using real world data sets by means of the proposed approach. The proposed approach is compared with other methods in literature for performance. Finally, the paper is summarized with conclusions.

Methodology

Predictive Model: TS-CNFS

The kernel of prediction is a Takagi-Sugeno complex neuro-fuzzy system (TS-CNFS) used as the predictive model. When expanded, it contains six neural-network layers and is illustrated in Figure 1. These are the layers of input, complex fuzzy sets, activation, normalization and output, respectively. Each layer is formed by artificial neurons (termed *nodes* for simplicity). Nodes in each layer have specific functionality to be explained below.

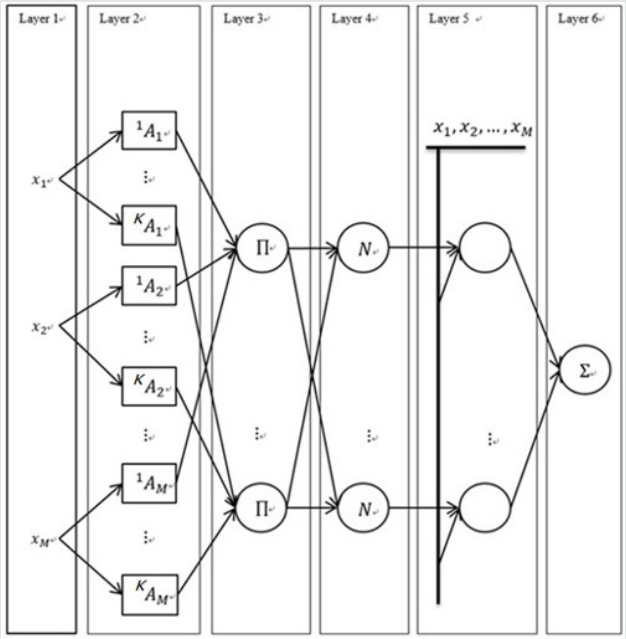

Figure 1: Illustration Of The Proposed Model.

Layer 1: Has *M* nodes in total corresponding to the inputs to the predictive model, respectively. The inputs are filtered (or selected) first by feature selection.

Layer 2: Contains nodes characterized by complex fuzzy sets. These nodes are termed as the complex fuzzyset nodes. Each input to the predictive model corresponds to a certain amount of nodes (that is, a group of nodes) in the layer. More specifically, the group size of nodes of one input can be different from that of another. Note that to form an if part (or termed a *premise*) with *M* conditions, each

input provides a node from its group of nodes in Layer 2 to make the *if* part. Note that an *if* part is regarded as a node in Layer 3. Each of the *M* conditions of an *if* part can be expressed by a simple proposition with the form "v is A," where v is a linguistic variable and A is a linguistic value. As a result, an *if* part is expressed by a complex proposition in the form of statement "v_1 is A_1 and ... and v_M is A_M ", where each of the *M* inputs is regarded as a linguistic variable with its value. For the ith linguistic variable v_i, there is a *term set* symbolized as T_i, whose size (cardinality) is denoted as $|T_i|$, containing a group of linguistic values The term set T_i is expressed as follows.

$$T_i = \left\{ A_{i,j},\ j = 1,2,\ldots,|T_i| \right\}, \tag{1}$$

where $A_{i,j}$ is the jth linguistic value of the ith variable. Accordingly, layer 2 has *M* groups of nodes corresponding to *M* term sets of linguistic values, respectively. Consequently, there are $\sum_{i=1}^{M} = 1|T_i|$ nodes in layer 2. For the linguistic value $A_{i,j}$, a node is characterized mathematically using a complex fuzzy set. A complex fuzzy set is an advanced fuzzy set whose complex-valued membership function μ is given as $\mu(x) = r(x)\cdot\exp\left(j\omega(x)\right)$, where $r(x)$ is the amplitude function with the restriction $0 \le r(x) \le 1; \omega(x)$ is the phase function with the restriction $\omega(x) \in R; j^2 = -1; x$ is a generic numerical variable. This paper presents the following membership function design for complex fuzzy sets.

$$\begin{aligned} &\mu(x;c,\sigma,\lambda) = r(x;c,\sigma)\cdot\exp\left(j\omega(x;c,\sigma,\lambda)\right), \\ &r(x;c;\sigma) = \exp\left(-0.5\left(\frac{x-c}{\sigma}\right)^2\right), \\ &\omega(x;c,\sigma,\lambda) = -r(x;c,\sigma)\cdot\left(\frac{x-c}{\sigma^2}\right)\cdot\lambda, \end{aligned} \tag{2}$$

where {c, σ, λ} are the parameters of {center, spread, phase factor} of a complex fuzzy set. The class of complex fuzzy sets defined by equation (2) is termed as *Gaussian complex fuzzy sets* [17], for their membership functions share the same basis using a Gaussian function and its first derivative with respect to x. In general, $\mu(x;c;\sigma,\lambda)$ is a complex-valued function, which can be expressed as $\mu(x;c;\sigma;\lambda) = \mathrm{Re}\left((x;c;\sigma;\lambda)\right) + j\,\mathrm{Im}\left(\mu(x;c;\sigma;\lambda)\right)$, where $\mathrm{Re}(\cdot)$ and $\mathrm{Im}(\cdot)$ are the real and imaginary parts, respectively. For the complex fuzzy set of $A_{i,j}$, the complex-valued membership function is denoted as $\mu_{i,j}\left(x_i;c_{i,j};\sigma_{i,j};\lambda_{i,j}\right)$. The parameters $\left\{\left(c_{i,j};\sigma_{i,j};\lambda_{i,j}\right)\right\}, i = 1,2,\ldots,M$ and $j = 1,2,\ldots,|T_i|\}$ are termed as the set of *complex fuzzy set parameters*, which are to be optimized by the method of MCACO. For the prediction of multiple targets, the output of the node for $A_{i,j}$ in Layer 2 is arranged to be a vector, denoted as $\vec{\mu}_{i,j}$, whose components are derived from the membership information of $\mu_{i,j}\left(x_i;c_{i,j};\sigma_{i,j};\lambda_{i,j}\right)\cdot$ (denoted as $\mu_{i,j}$ for simplicity), for example real and imaginary parts as well as the $\mu_{i,j}$ itself, given below.

$$\vec{\mu}_{i,j} = \left[\mu_{i,j,1}, \mu_{i,j,2}, \ldots . \mu_{i,j,N}\right]^T, \text{ where } N \text{ is the vector size.} \tag{3}$$

Layer 3: Contains *K* nodes, each of which represents an *if* part with *M* conditions. An *if* part and a *then* part (or termed a *consequence*) can make an *if-then* rule. The nodes of the layer represent *K if* parts in total, and therefore *if-then* rules to make. The type of Takagi-Sugeno (TS) *if-then* rules is adopted in the paper. The kth TS rule (for k=1, 2, ..., K) can be expressed as follows.

$$\begin{aligned} &\text{IF } v_1 \text{ is } A_1^{(k)}(x_1) \text{ and } v_2 \text{ is } A_2^{(k)}(x_2) \text{ and } \ldots \text{and } v_M \text{ is } A_{(k)}^{M}(x_M) \\ &\text{THEN } y^{(k)} = a_0^{(k)} + a_1^{(k)}x_1 + \ldots + a_M^{(k)}x_M, \end{aligned} \tag{4}$$

where v_i is the ith input linguistic variable whose base (numerical) variable is denoted as x_i; $A_i^{(k)}$ is the linguistic value of v_i in the *if* part of the kth rule and is defined using a complex fuzzy set, whose complexvalued membership function is a function of x_i and is denoted as $\mu_i^{(k)}(x_i)$ or $A_i^{(k)}(x_i)$ for convenience. Note that the linguistic value $A_i^{(k)}$ is a linguistic value from the term set T_i given in equation (1). For the *then* part of equation (4), it is a TS linear function of the input base variables {x_i, i = 1,2, ... , M}, whose parameters $\left\{a_i^{(k)}, i = 0,1,\ldots M\right\}$ are called the *consequent* parameters of the kth rule, which are to be optimized by the method of RLSE. For the complex fuzzy set of $A_i^{(k)}$, using equation (2), the membership function is denoted as $\mu_i^{(k)}\left(x_i;c_i^{(k)},\sigma_i^{(k)},\lambda_i^{(k)}\right)$, given below.

$$\mu_i^{(k)}\left(x_i;c_i^{(k)},\sigma_i^{(k)},\lambda_i^{(k)}\right) = r\left(x_i;c_i^{(k)},\sigma_i^{(k)}\right)\cdot\exp\left(j\omega\left(x_i;c_i^{(k)},\sigma_i^{(k)},\lambda_i^{(k)}\right)\right), \tag{5}$$

where $r\left(x_i;c_i^{(k)},\sigma_i^{(k)}\right) = \exp\left(-0.5\left(\frac{x_i - c_i^{(k)}}{\sigma_i^{(k)}}\right)^2\right)$ and

$$\omega\left(x_i;c_i^{(k)},\sigma_i^{(k)},\lambda_i^{(k)}\right) = -r\left(x_i;c_i^{(k)},\sigma_i^{(k)}\right)\cdot\left(\frac{x_i - c_i^{(k)}}{\left(\sigma_i^{(k)}\right)^2}\right)^{(k)}\cdot\lambda_i^{(k)}; \left\{c_i^{(k)},\sigma_i^{(k)},\lambda_i^{(k)}\right\}$$

are the parameters of the complex fuzzy set of $A_i^{(k)}$. For multi-target prediction, a vector based on $\mu_i^{(k)}\left(x_i;c_i^{(k)},\sigma_i^{(k)},\lambda_i^{(k)}\right)$ given in (5) is arranged as follows.

$$\vec{\mu}_i^{(k)}\left(x_i;c_i^{(k)},\sigma_i^{(k)},\lambda_i^{(k)}\right) = \left[\mu_{i,1}^{(k)}, \mu_{i,2}^{(k)}, \ldots, \mu_{i,N}^{(k)}\right]^T, \tag{6}$$

where $\mu_i^{(k)} \equiv \mu_i^{(k)}\left(x_i;c_i^{(k)},\sigma_i^{(k)},\lambda_i^{(k)}\right); \mu_{i,q}^{(k)}$ denotes the qth component of the vector $\vec{\mu}_i^{(k)}$. Note that

$\left[\mu_{i,1}^{(k)}, \mu_{i,2}^{(k)}, \ldots, \mu_{i,N}^{(k)}\right]^T$ can be arranged using the information by $\mu_i^{(k)}$, for example $\left[\mu_{i,1}^{(k)}, \mathrm{Re}\left(\mu_i^{(k)}\right), \mathrm{Im}\left(\mu_i^{(k)}\right)\right]^T$ (for N=3 in the case) or other possible forms, depending on applications. We

apply the *fuzzy-and* operation, using the "product" implementation, to aggregate the propositions of the *if* part of the *k*th rule in [4].

The output of the *k*th node of layer 3, which is the *activation level vector* of the *k*th *if* part (or regarded as the rule's *firing strength vector* for multiple outputs), is given in a vector form below.

$$\vec{\beta}^{(k)} = \left[\beta_1^{(k)}, \beta_2^{(k)}, \ldots \beta_N^{(k)} \right]^T, \tag{7}$$

where the *q*th component of $\vec{\beta}^{(k)}$ is given as $\beta_q^{(k)} = \prod_{i=1}^{M} \mu_{i,q}^{(k)}$ (for *q*=1,2,...,*N*). Note that the vector $\vec{\beta}^{(k)}$ is a complex vector due to part of its components are complex.

Layer 4: Performs the normalization of $\left\{ \vec{\beta}^{(k)}, k = 1, 2, \ldots, K \right\}$. There are *K* nodes performing the normalization in this layer, where each node receives all $\left\{ \vec{\beta}^{(k)}, k = 1, 2, \ldots, K \right\}$ from layer 3. The output of the *k*th node is in a vector form given below.

$$\vec{\rho}^{(k)} = \left[\rho_1^{(k)}, \rho_2^{(k)}, \ldots \rho_N^{(k)} \right]^T, \tag{8}$$

$$\vec{\rho}_q^{(k)} = \frac{\beta_q^{(k)}}{\sum_{k=1}^{K} \beta_q^{(k)}} \text{ for } q=1,2,\ldots N,$$

where $\rho_q^{(k)}$ the *q*th component of $\vec{\rho}^{(k)}$. The $\vec{\rho}^{(k)}$ is a complex vector for all the components are complex after the normalization of $\vec{\beta}^{(k)}$.

Layer 5: Performs the TS *then* parts. There are *K* nodes for such a purpose in this layer. The output of the *k*th node is in a vector form given below.

$$\vec{\hat{y}}^{(k)} = \left[\hat{y}_1^{(k)}, \hat{y}_2^{(k)}, \ldots \hat{y}_N^{(k)} \right]^T,$$

$$\vec{\hat{y}}_q^{(k)} = \rho_q^{(k)} \left\{ \left(\vec{x}_a \right)^T \vec{a}^{(k)} \right\} \text{ (for } q=1, 2, \ldots, N),$$

$$\vec{a}^{(k)} = \left[a_0^{(k)}, a_1^{(k)}, \ldots, a_M^{(k)} \right]^T \tag{9}$$

$$\vec{x}_a = \left[1, x_1, \ldots, x_M \right]^T$$

where $\vec{x}_a$ is called the *augmented input vector* and $\vec{a}^{(k)}$ is the consequent parameter vector of the TS *then* part of the *k*th rule. With the method of RLSE, the set of parameters of $\left\{ a_i^{(k)}, k = 1, 2, \ldots, K \right\}$ in equation (9) are to be optimized.

Layer 6: Summarizes the outputs of layer 5 to obtain the vector of multiple model outputs, given below.

$$\vec{\hat{y}} = \sum_{k=1}^{K} \vec{\hat{y}}^{(k)} = \left[\hat{y}_1, \hat{y}_2, \ldots \hat{y}_N \right]^T,$$

$$\vec{\hat{y}}_q = \sum_{k=1}^{K} \vec{\hat{y}}_q^{(k)} = \sum_{k=1}^{K} \rho_q^{(k)} \left\{ \left(\vec{x}_A \right)^{T \vec{a}(k)} \right\} \text{ for } q=1,2,\ldots,N, \tag{10}$$

where $\hat{y}_q$ is the *q*th component of the model output vector $\vec{\hat{y}}$. As shown in equation (10), the *q*th model output component $\hat{y}_q$ is complex due to $\rho_q^{(k)}$ and thus can be expressed as $\hat{y}_q = \mathrm{Re}\left(\hat{y}_q \right) + j\, \mathrm{Im}\left(\hat{y}_q \right)$, which can be applied to dual-target applications. Note that the model output vector $\vec{\hat{y}}$ can theoretically be applied to applications with 2*N* real-valued targets.

The flowchart of the prediction system is shown in Figure 2. From left to right, there are three sub-flowcharts: the method MCACO, the T-S complex neuro-fuzzy system and the method RLSE. The whole procedure starts at the first step of the MCACO and ends by the last step of the MCACO. Multiple ant colonies of MCACO in a cooperative way search for the solution of *complex fuzzy set parameters* in layer 2 of TS-CNFS. The RLSE method is responsible for the solution of TS *consequent parameters* in layer 5 of TS-CNFS. And, the MCACO and RLSE methods work together concordantly as one method, so to be named the MCACO-RLSE method. Each ant of the multiple ant colonies is evaluated by its performance in the machine learning process. A *performance* (or termed *cost*) criterion which is used to evaluate every ant is an error index. The training data set (Ω) for the machine learning of the proposed predictive model is expressed below [17-48].

$$\Omega = \left\{ \left(\vec{x}^{\{i\}}, \vec{y}^{\{i\}} \right), i = 1, 2, \ldots n \right\},$$

$$\vec{x}^{\{i\}} = \left[x_1^{\{i\}}, x_2^{\{i\}}, \ldots, x_N^{\{i\}} \right]^T, \tag{11}$$

$$\vec{y}^{\{i\}} = \left[y_1^{\{i\}}, y_2^{\{i\}}, \ldots, y_N^{\{i\}} \right]^T,$$

where {*i*} indicates the *i*th pair of Ω; $\left(\vec{x}^{\{i\}}, \vec{y}^{\{i\}} \right)$ is the *i*th data pair of (input, target); $\vec{x}^{\{i\}}$ is the input vector of the *i*th pair and $\vec{y}^{\{i\}}$ is the target vector; *n* is the size of Ω. The performance of each ant in the MCACO is evaluated by the mean square error (MSE), the so-called cost function, defined below.

$$MSE = \sum_{i=1}^{n} \left(\vec{e}^{\{i\}} \right)^* \vec{e}^{\{i\}}, \tag{12}$$

$$\vec{e}^{\{i\}} = \left[e_1^{\{i\}}, e_2^{\{i\}}, \ldots, e_N^{\{i\}} \right]^T = \vec{y}^{\{i\}} - \vec{\hat{y}}^{\{i\}}$$

where $e_q^{\{i\}}$ is the *q*th component error (*q*=1,2,...,*N*) between the target vector $\vec{y}^{\{i\}}$ and the model output vector $\vec{\hat{y}}^{\{i\}}$ given in equation [10] when the *i*th training data pair is presented to the predictive model; $\left(\vec{e}^{\{i\}} \right)^*$ indicates the complex conjugate of $\vec{e}^{\{i\}}$. The MCACO-RLSE method is specified in detail in the following.

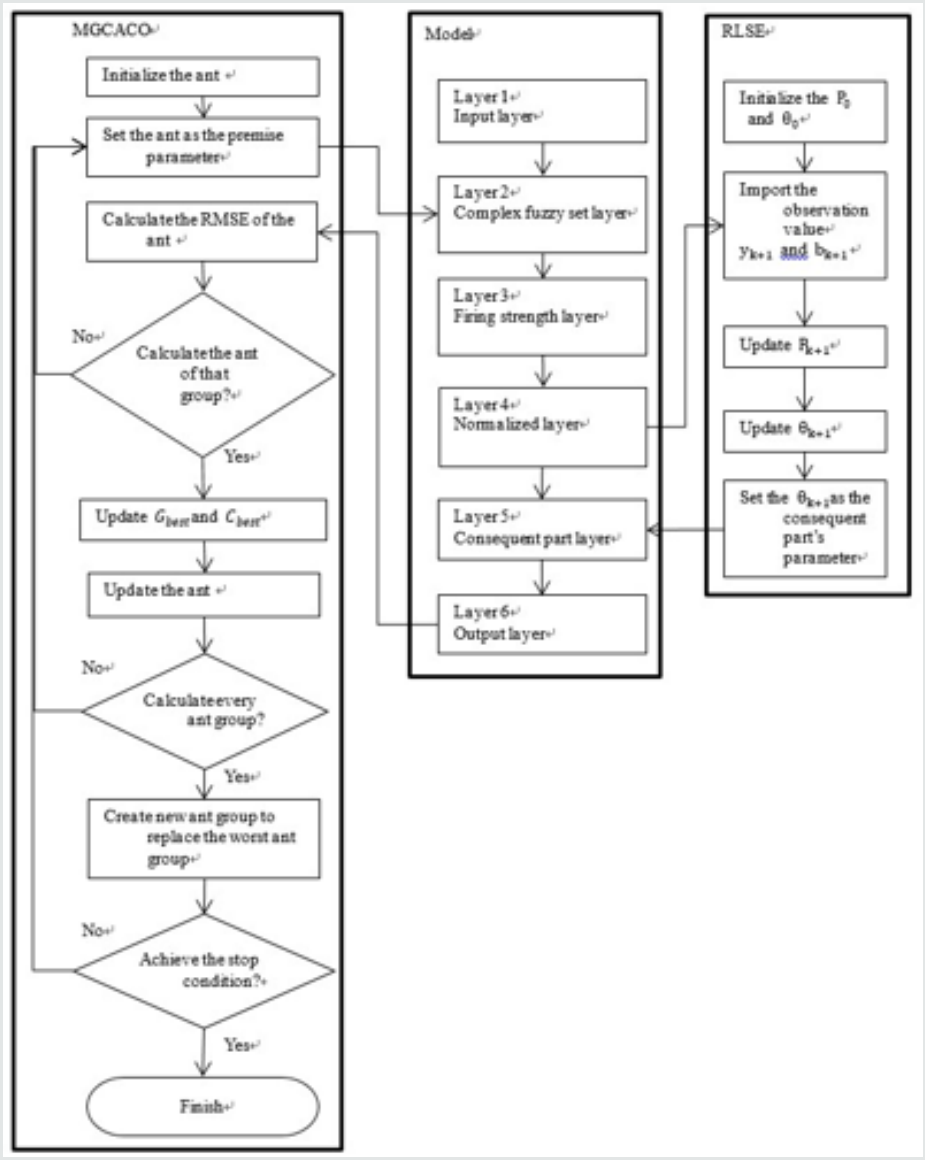

Figure 2: Flowchart of the Proposed MCACO-RLSE Learning Method.

Machine Learning Method: MCACO-RLSE

Multi-swarm continuous ant colony optimization

In the paper, the problem under discussion is to minimize an objective function f(X) without constraints. The variable X is a parameter set used in layer 2 of the prediction system, i.e. every parameter set $\{m, \sigma, \lambda\}$ of all used complex Gaussian membership functions. Function f(X) evaluates the root-mean-squared error between the predicted and the expected when the current variable X is used. The method MCACO is proposed based on the method continuous ant colony optimization (CACO) [26]. However, the fore uses multiple ant colonies and an ant in different colony represents different part of the whole parameter set X. Thus, a whole parameter set is divided into several small sets so that every colony searches only in a small solution space. This is more effective than searching a whole large solution space.

Mechanism of generating new ants: Different colony searches for different part of the whole parameter set. Thus, only ants of the same colony are used to generate their own new ants. Initially, all ant colonies are randomly generated where every ant in different colony represents a different part of parameter vector X. When all ants in a colony have been evaluated, a key ant is randomly selected to generate new ants. The key ant decides how a Gaussian function of weighting probability is distributed. Based on these Gaussian distributions, new ants are therefore generated by contiguously recombining the parameters selected one by one from current ant colony. The mechanism of selecting key ant is like the way of playing roulette wheel. In current ant colony, all ants are ordered according to their objective function values f(X) from small to large. In the ant sequence, the *l*th ant's weighting can be obtained as below.

$$\omega_l = \frac{1}{qh\sqrt{2\pi}} \exp\left(\frac{-(l-1)^2}{2q^2h^2}\right). \quad (13)$$

There are total h ants in the colony and q is a learning parameter between 0 and 1. The larger the weighting ω_l is the better the solution that the lth ant represents is. The selection probability is therefore calculated based on equation (8).

$$\rho_l = \frac{\omega_l}{\sum_{r=1}^{h} w_r}. \tag{14}$$

Assume the lth ant is selected as the key ant. Thus, its ith parameter is used as the averaged value μ_l^i and the corresponding standard deviation is shown as below.

$$\sigma_l^i = \xi \sum_{e=1}^{h} \frac{s_e^i - s_l^i}{h-1}. \tag{15}$$

The parameter ξ is an evaporation rate. Of the ith parameter, the Gaussian function of weighting probability $G^i(x)$ is distributed as follows.

$$G^i(x) = \omega_l \frac{1}{\sigma_l^i \sqrt{2\pi}} \exp\left(\frac{-\left(x-\mu_l^i\right)^2}{2\sigma_l^{i2}}\right). \tag{16}$$

In words, different parameters have different distributions of weighting probability. A new ant is therefore created by recombining the parameters selected one by one.

Three strategies of ant evolution: Ants in different colonies represent different parts of the whole parameter set. Three strategies are therefore introduced to evaluate ants. By these strategies, all ant colonies are iteratively renewed to improve machine learning.

a. Evaluation strategy: An ant can be evaluated only if it becomes a whole parameter set. By the evaluation strategy, every ant in some colony is evaluated when combined with every the best ant in the other colonies. Initially, a randomly selected ant is used when the colony haven't been evaluated yet. The strategy helps effective machine learning.

b. Elimination strategy: The elimination strategy is to totally renew the worst colony in where the best ant is the worst among all. Such colony will take time in machine learning. However, it may contain potential ants. The way to renew the whole worst colony is to generate sufficient new ants using the current ants and then replace all the latter.

c. Inheritance strategy: Ants in colonies perform well, i.e. with small root mean-squared error, because of containing good parameters. The inheritance strategy tries to keep these good parameters for next machine learning. Therefore, in order to renew such a colony, a specific number of top ants are kept while the rest are replaced by newly generated ants.

The flowchart of method MCACO is shown in Figure 3. The first step is to specify the parameters of MCACO, i.e. the number of total iterations, the number of total ant colonies, the number of top ants to be kept, the number of top best ants in every colony to be collected into the generation pool etc. Then, randomly generate ants as the first generation of all colonies. The evaluation to an ant is a root mean-squared error between the expected and the predicted when the ant combined with every the best of rest colonies is applied to the prediction system. When all ants of all colonies have been evaluated, all current colonies are then renewed and replaced by the three strategies until the number of total iterations has been reached.

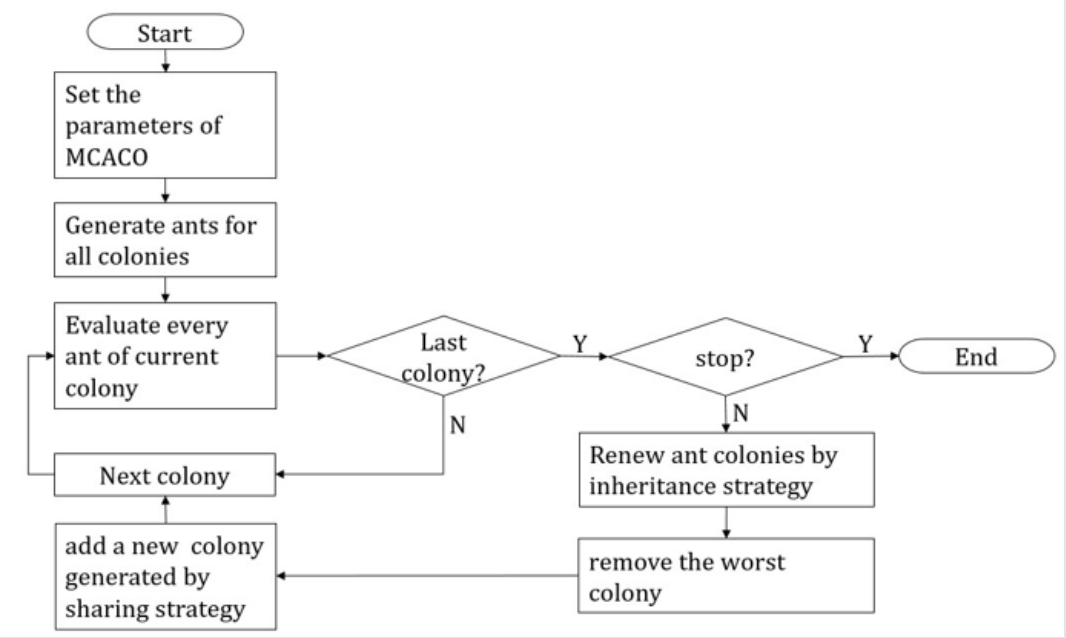

Figure 3: Flowchart of the MCACO Method.

Recursive least squares estimation (RLSE) for multiple targets

The RLSE method devotes to estimate all consequent parameters $\left\{a_i^{(k)}, i=0,1,\ldots,M \text{ and } k=1,2,\ldots,K\right\}$ of the predictive model for multiple targets. Based on the method of least squares estimation, the RLSE method for the problem of linear regression optimization is efficient. A general least squares estimation problem is specified by a linearly parameterized expression given as follows.

$$y=\theta_1 f_1(x)+\theta_2 f_2(x)+\ldots+\theta_m f_m(x)+e, \tag{17}$$

where x is the input to model; y is the target; $\left\{f_i(x), i=1,2,\ldots,m\right\}$ are known functions of x; $\left\{\theta_i, i=1,2,\ldots,m\right\}$ are the parameters to be estimated; e is the error. With the training data set (Ω), we have the equation set given below.

$$\left[\vec{y}^{\{1\}}\right]_{N\times 1}=\left[f_1\left(\vec{x}^{\{1\}}\right)\right]_{N\times 1}\theta_1+\left[f_2\left(\vec{x}^{\{1\}}\right)\right]_{N\times 1}\theta_2+\ldots+\left[f_m\left(\vec{x}^{\{1\}}\right)\right]_{N\times 1}\theta_m+\left[\vec{e}^{\{1\}}\right]_{N\times 1},$$

(18)

$$\left[\vec{y}^{\{n\}}\right]_{N\times 1}=\left[f_1\left(\vec{x}^{\{n\}}\right)\right]_{N\times 1}\theta_1+\left[f_2\left(\vec{x}^{\{n\}}\right)\right]_{N\times 1}\theta_2+\ldots+\left[f_m\left(\vec{x}^{\{n\}}\right)\right]_{N\times 1}\theta_m+\left[\vec{e}^{\{n\}}\right]_{N\times 1}$$

In matrix notation, (18) can be rewritten as follows.

$$A=A\theta+E, \tag{19}$$

Where,

$$A=\begin{bmatrix}\left[f_1\left(\vec{x}^{\{1\}}\right)\right] & \left[f_2\left(\vec{x}^{\{1\}}\right)\right] & \cdots & \left[f_m\left(\vec{x}^{\{1\}}\right)\right]\\ \vdots & \vdots & \ddots & \vdots\\ \left[f_1\left(\vec{x}^{\{n\}}\right)\right] & \left[f_2\left(\vec{x}^{\{n\}}\right)\right] & \cdots & \left[f_m\left(\vec{x}^{\{n\}}\right)\right]\end{bmatrix}_{(N\cdot n)\times m} \tag{20}$$

$$\theta=\begin{bmatrix}\theta_1\\ \theta_2\\ \vdots\\ \theta_m\end{bmatrix}_{m\times 1}, \tag{21}$$

$$y=\begin{bmatrix}\left[\vec{y}^{\{1\}}\right]\\ \left[\vec{y}^{\{2\}}\right]\\ \vdots\\ \left[\vec{y}^{\{n\}}\right]\\ \\ \\ \end{bmatrix}_{(N-n)\times 1} \tag{22}$$

$$E=\begin{bmatrix}\left[\vec{e}^{\{1\}}\right]\\ \left[\vec{e}^{\{1\}}\right]\\ \vdots\\ \left[\vec{e}^{\{n\}}\right]\\ \\ \\ \end{bmatrix}_{(N-n)\times 1}. \tag{23}$$

The consequent parameters of the predictive model can be arranged to be a vector that is regarded as the vector θ. For the recursive process of RLSE, a training data pair of Ω is presented at each *recursive iteration*. The next iteration follows based on the result of the previous one. The process goes on until all data pairs of Ω have been presented to finish one *recursive epoch*. The optimal estimate of θ, in terms of the minimum sense of $\|E\|$, can be obtained using the training data pairs of Ω recursively by the following RLSE equations.

$$P(i+1)=P(i)-\left\{\frac{P(i)B(i+1)}{I_{N\times N}+\left(B(i+1)\right)^T P(i)B(i+1)}\right\}\left(B(i+1)\right)^T P(i), \tag{24}$$

$$\theta(i+1)=\theta(i)+P(i+1)B(i+1)\left\{\vec{y}^{\{i+1\}}-\left(B(i+1)\right)^T\theta(i)\right\}, \tag{25}$$

for i = 0,1, … , (n − 1), where the index i indicates the ith recursive iteration; n is the number of training data pairs of Ω; N is the size of $\vec{y}^{\{i+1\}}$; $\vec{y}^{\{i+1\}}$ is the target vector of the (i + 1)th data pair; **P**(i) is a m-by$m$ projection matrix at the ith recursive iteration; $I_{N\times N}$ denotes the N-by-N identity matrix; initially **P**(0) = $\alpha I_{m\times m}$; α is a very large positive value and $I_{m\times m}$ is the m-by-m identity matrix. The transpose of

(**B**(i + 1)) in the RLSE equations is the ith row block of **A**, that is,

$$B(i+1)^T=\left[\left[f_1\left(\vec{x}^{\{i+1\}}\right)\right],\left[f_2\left(\vec{x}^{\{i+1\}}\right)\right],\ldots,\left[f_m\left(\vec{x}^{\{i+1\}}\right)\right]\right]_{N\times m}. \tag{26}$$

Note that the RLSE given above is to deal with multiple targets simultaneously and recursively.

For the implementation of RLSE on the predictive model with the training data set Ω in the learning

T process, the (**B**(i + 1)) is given below.

$$\left(B\left(i+1\right)^{T}\right)=\left[b^{(1)}\left(i+1\right),b^{(2)}\left(i+1\right),\ldots,b^{(K)}\left(i+1\right)\right]_{N\times\left(K\cdot\left(M+1\right)\right)}',$$

(27)

$$b^{(k)}\left(i+1\right)=\left[\vec{\rho}^{(k)}\left(i+1\right),\vec{\rho}^{(k)}x_1^{\{i+1\}},\ldots,\vec{\rho}^{(k)}x_M^{\{i+1\}}\right]_{N\times\left(M+1\right)}',$$

where the index k indicates the kth *if* part (k = 1,2, … , K) and the index (i + 1) indicates the (i + 1)th recursive iteration; $\vec{\rho}^{(k)}$ is specified in equation [8]; $x_j^{\{i+1\}}$ is the jth component (j=1,2,...,M)) of the input vector given in (11); {M, N, K} are the numbers of inputs, outputs and *if-then* rules of the predictive model,

T respectively. Note that when equation (26) compares with (27) for the dimension of ($\mathbf{B}(i+1)$) , the equation $m=K\cdot\left(M+1\right)$ holds. And, the vector of consequent parameters of the predictive model is given as follows.

$$\theta(i)=\begin{bmatrix}\vec{a}^{(1)}(i)\\ \vec{a}^{(2)}(i)\\ \vdots\\ \vec{a}^{(K)}(i)\end{bmatrix}_{\left(K\cdot\left(M+1\right)\right)\times 1},\quad (28)$$

where $\vec{a}^{(k)}\left(i\right)$ is the consequent parameter vector of the kth TS *then* part (k=1,2,...,K) at the ith recursive iteration of RLSE. The optimal estimate for θ is $\theta(n)$, where n denotes the final recursive iteration when one RLSE epoch is done. The procedure of the proposed MCACO-RLSE is summarized as follows.

Step 1. Initialize the setting of MCACO-RLSE. Start the MCACO.

Step 2. Calculate membership degree vectors of complex fuzzy sets in layer 2. Obtain the firing strength vectors in layer 3 and do the normalization in layer 4 for these vectors of the predictive complex neuro-fuzzy model.

Step 3. Update the consequent parameters in layer 5 of the model by the RLSE.

Step 4. Calculate forecast vector by the model and obtain error vector for each training data pair until all data pairs have been presented. Get the cost of ant in MSE.

Step 5. Calculate the cost of each ant of MCACO by repeating Steps 2 to 4 until all ants are done. Obtain the best ant position so far for each of the multiple swarms. Combine the positions of the best ants of the ant swarms together to get the solution so far to update the parameters of complex fuzzy sets in layer 2 of the predictive model.

Step 6. Check for any of stopping conditions to stop. Otherwise go back to Step 2 and continue the procedure.

Feature Selection Crossing Multiple Targets

Feature selection is a method of selecting a compact set of important features out from candidate features. These selected features are used as the inputs to the prediction system with multiple targets. In the paper, the inputs and outputs of the prediction system are difference values of real observations. For multiple target prediction, a two-stage feature selection is proposed, where useful features for each single target are determined explicitly in the first stage and, based on the determined in stage 1, those features significant to all targets are selected in the second stage. Suppose that there are a set of candidate features to be selected in the procedure of feature selection in which multiple targets are considered. Let the candidate features be placed in a pool called the *candidate feature pool* denoted as C_p, given below.

$$C_p=\left\{X_k,k=1,2,\ldots,\left|C_p\right|\right\},\quad (29)$$

where X_k is the kth candidate feature of C_p; $\left|C_p\right|$ denotes the size of C_p. The multiple targets considered in the feature selection procedure is given as follows.

$$T=\left\{Y_i,i=1,2,\ldots,\left|T\right|\right\},\quad (30)$$

where Y_i is the ith target in the set T whose size is denoted as $\left|T\right|$. All the candidate features of C_p and the targets of T are regarded as random variables.

Stage 1 of Feature Selection

Assume there is a target denoted as Y from the target set T. Let $H(Y)$ denote the entropy of target Y, where $H\left(Y\right)=-\int_{\upsilon(y)}p\left(y\right)\log p\left(y\right)dy$; y is a generic event of Y; $U(y$) is the universal event set for y ;

$p(y)$ is the probability density distribution of y . From the candidate feature pool C_p , a feature X is considered. Let the feature X be separated into two parts X^+ and X^- , where X^+ and X^- denote the positive and negative parts of X, respectively. That is, the values of X^+ are positive and those of X^- are non-positive. Let $H_X+(Y)$ denote the entropy of target Y conditioned by feature X^+ . According to the Shannon's information theory [11], the mutual information between Y and X^+ is given as $I(X^+, Y) = H(Y) - H_X+(Y)$. Now another idea of information called the *influence information* is defined for the influence made by feature X to target Y. Such influence information is denoted as $I_{x\to y}$ given below.

$$I_{X\to Y}=I\left(X^{+},Y\right)\int_{\circ}^{\infty}p\left(x\right)dx+I\left(X^{-},Y\right)\int_{-\infty}^{\circ}p\left(x\right)dx,\quad (31)$$

where $p(x)$is the probability density distribution of x, a generic event of feature X. Note that the distributions of $p(x)$ and $p(y)$ can be estimated by density estimation [44]. Note that the integration

$\int_{\circ}^{\infty} p(x)dx$ indicates the probability of $I(X^+, Y)$ and $\int_{-\circ}^{\infty} p(x)dx$ the probability of $I(X^-, Y)$. In contrast to the symmetry of Shannon's mutual information, the influence information given in (31) is asymmetric,

that is, $I_{x\to y} \neq I_{y\to x}$.

The first stage procedure of feature selection is given below.

Step 1. Prepare each target of T a pool to contain features to be selected from C_p. For target Y_i of T, the pool is termed the *selected features pool* for it, denoted as S_{Yi}, for $i = 1,2,...,|T|$. Initially S_{Yi} is empty.

Step 2. Let C_{Yi} be the pool containing all the candidate features for a target Y_i, that is, $C_{yi} = C_p = \{X_k, k = 1,2,...,\{|C_p|\}\}$.

Step 3. Calculate the mutual information of Y_i, to which compute influence information of each candidate feature of C_{Yi}. For each candidate feature, calculate *selection gain*. The selection gain of X_k of C_{Yi} making contribution to target Y_i is defined as follows.

$$G_{X_k\to Y_i} = I_{X_k\to Y_i} - R_{X_k\to S_{Y_i}}, \tag{32}$$

$$R_{X_k\to S_{Y_i}} = \frac{1}{2|S_{Y_i}|}\sum_{j-1}^{|S_{Y_i}|}\left(I_{X_k\to S_{Y_i}(j)} + I_{S_{Y_i}(j)\to X_k}\right),$$

for k = 1,2, ... , $|C_{yi}|$, i = 1,2, ... , $|T|$, and j = 1,2, ... , $|S_{yi}|$, where $G_{Xk\to Y=Yi}$ is the selection gain when selecting feature X_k into the pool S_{Yi} for target Y_i; $R_{Xk\to Y=SYi}$ is an averaged redundant information made by feature X_k to all selected features already in the pool S_{Yi} so far; $S_{Yi}(j)$ indicates the jth feature already in the pool S_{Yi} so far; $|S_{yi}|$ is the size of S_{Yi} so far. Note that the selection gain indicates the measure of significance for a random variable making influence/contribution to another.

Step 4. Check for the candidate feature whose selection gain is the largest over all other ones in C_{Yi}. If it is positive, update the pools C_{Yi} and S_{Yi}, as follows.

$$C_{Yi} = C_{Yi} \ominus X_m, \tag{33}$$

$$S_{Y_i} = S_{Y_i} \oplus X_m,$$

where X_m indicates the candidate feature with the positive max selection gain over all other ones in C_{Yi} so far; $C_{Yi} \ominus X_m$ means the operation of excluding X_m from C_{Yi} and $S_{yi} \oplus X_m$ means the operation of including X_m into S_{Yi}. Otherwise, go to step 6.

Step 5. If $|C_{yi}| = 0$, go to step 6. Otherwise, go back to step 3.

Step 6. If $i = |T|$, update $i = i + 1$ to get next target and go back to step 2. Otherwise, stop the procedure and output all the selected feature pools $\{i = 1,2,...,|T|\}$, where $S_{yi} = \{S_{yi}(j), j = 1,2,...|S_{yi}|\}$ and $S_{yi}(j)$ is the jth selected feature in S_{Yi} for target Y_i.

Stage 2 of Feature Selection

For each target in the target set T, the result has been done with the first stage of feature selection, as shown in $\{S_{Yi}, i = 1,2,...,|T|\}$, based on which the second stage of feature selection goes on for further consideration of multiple targets. For $\{S_{Yi}, i = 1,2,...,|T|\}$, the contribution by each of all the selected features to all targets will be inspected separately. Thus, the features that are globally important to all targets can be decided. The feature selection procedure of stage 2 is given below.

Step 1. Identify exclusively all the features that are in the pools $\{S_{Yi}, i = 1,2,...,|T|\}$.

These features are placed into a set denoted as Ω = $\{\phi_k k = 1,2,...|\Omega|\}$ where ϕ_k is the kth feature of Ω.

Step 2. Compute coverage rate of each feature in Ω to all the targets $\{Y_i, i = 1,2,...,|T|\}$.

Compute the count of ϕ_k in the pools $\{S_{Yi}, i = 1,2,...,|T|\}$, denoted as $n_{OL}(\phi_k)$. And, compute the *coverage rate* of ϕ_k to all the targets, denoted as $\omega(\phi_k)$, given below.

$$\omega(\phi_k) = \frac{n_{OL}(\phi_k)}{|T|}, \tag{34}$$

for $k = 1,2,...,|\Omega|$, where $|\Omega|$ denotes the size of Ω. Compute $\bar{\omega} = \frac{1}{|\Omega|}\sum_{k=1}^{|\Omega|}\omega(\phi_k)$.

Step 3. Compute contribution index of each feature in Ω to all the targets.

Compute the *contribution index* by ϕ_k to all the targets, denoted as $\rho(\phi_k)$, given below.

$$\rho(\phi_k) = \omega(\phi_k)\cdot G_{sum}(\phi_k), \tag{35}$$

for $k = 1,2,...,|\Omega|$, where $G_{sum}(\phi_k) = \sum_{i=1}^{|T|} G_{\phi_k\to Y_i}$. Compute $\bar{G}_{sum} = \frac{1}{|\Omega|}\sum_{k=1}^{|\Omega|} G_{sum}(\phi_k)$.

Step 4. Test the features in Ω for the following condition.

If $\rho(\phi_k) > \rho th$, then $n_{tmp} = n_{tmp+1}$, for $k = 1,2,...,|\Omega|$, where $\rho th = \bar{\omega}\cdot G_{sum}$ is termed the condition threshold; n_{tmp} records the number of feature variables in Ω fulfilling the condition above. Note that the condition threshold ρth can also be set in other ways.

Step 5. Decide the finally selected features.

Set lower and upper limits for finalizing the number of feature variables. These features are placed finally in the so-called final

pool (F_P). The lower and upper limits are denoted as n_L and n_U, respectively. The number of features in F_P is denoted as n_F or $|F_p|$. Note that n_L and n_U are set depending on application. The n_{tmp} is obtained in step 4.If $n_{tmp} < n_L$, then $n_F = n_L$, elseif $n_L \le n_{tmp} \le n_U$, then $n_F = n_{tmp}$, else $n_F = n_U$, end. After ranking $\{\rho(\phi_k), k = 1,2,...,|\Omega|\}$, select top n_F corresponding features $\{f_i, i = 1,2, \ldots, n_F\}$ and place them into the final pool F_P, given as follows.

$$F_P = \{f_i, i = 1,2,...n_F\}, \quad (36)$$

where f_i is the ith feature of F_P contributing to all the targets given in the set T.

Results and Discussion

Table 1: Model Setting.

Number of inputs	3
Number of complex fuzzy sets per input	3
Type of fuzzy sets for premises	Gaussian complex fuzzy set

Table 2: MCACO-RLSE Setting.

MCACO	
Number of iterations	50
Number of ant colonies	3
Population per ant colony	30
Number of top best ants per colony	5
Number of renewal ants per colony per iteration	20
RLSE	
Number of consequence parameters	108
P(0)	αI
α	1×10^{10}
I	108-by-108 identity matrix
θ(0)	108-by-1 zero vector

Real world stock-market data are utilized to test the proposed approach. In experiment 1, the proposed system is to make dual outputs at the same time to predict the Taiwan capitalization weighted stock index (TAIEX) and the SSE composite index (SSECI) posted by Shanghai stock exchange. Experiment 2 is to make triple outputs simultaneously for prediction of three stock indexes, i.e. SSECI, the Standard & Poor's 500 (S&P 500) and Dow Jones industrial average (Dow Jones). Table 1 shows the model setting of our prediction system. The number of model outputs depends on how many targets to be predicted. For experiment 1, two real-valued targets (TAIEX and SSECI) are placed into the real and imaginary parts of a complex-valued target, respectively, to which the TS-CNFS needs only one complex-valued output to make prediction. For experiment 2 with three real-valued targets, the TS-CNFS model may need two complex-valued outputs, corresponding to two complex-valued targets. The first two of the three real-valued targets (SSECI, S&P 500 and Dow Jones) are placed into one complex-valued target and the third one is placed into the real part of the other complex-valued target whose imaginary part is placed with zero. Table 2 shows the parameters of approach MCACO-RLSE used for all the experiments.

Experiment 1: The prediction of TAIEX and SSECI is carried out by the proposed approach. All gathered data of close prices cover the whole year 2006. The data of the first ten months are used for training and those of the last two months are used to test and evaluate the performance of the prediction system. In the case, 15 features for each target are gathered so that there are total 30 candidate features. By the proposed feature selection, three features are selected as inputs to the prediction system. The first feature is selected from the features of TAIEX and the rest two from SSECI. Thus, the prediction system has total 27 TakagiSugeno if-then rules with one complex output. Table 3 shows performance comparison in terms of root mean square error by the proposed approach (in test phase) to nine compared methods in literature. The dual predictions during testing phase are outstanding. As shown in table 3, by our prediction system, both the predictions of TAIEX and SSECI have improved with 6% and 8% when compared with those obtained by Zhang et al. [36]. The performance by the proposed approach performing prediction of dual targets is superior to the compared methods in literature, even each of the compared is for prediction of one target only. The proposed prediction system demonstrates outstanding dual-predictions by using one complex valued model output for two real-valued targets. Moreover, as shown in this case the proposed MCACORLSE method has shown effective ability of machine learning through the empirical result. Furthermore, such successful performance only uses three inputs obtained by the proposed feature selection.

Table 3: Performance Comparison In Root Mean Square Error (Experiment 1).

Method	**TAIEX**	**SSECI**
Huarng et al.'s method [28]	82.8194	75.0643
Cheng et al.'s method [29]	99.7014	64.0943
Chen's method [30]	92.491	42.6612
Lee et al.'s method [31]	84.2257	82.0055
Egrioglu et al.'s method [32]	78.7521	114.9601
Wang et al.'s method [33]	222.9535	379.5415
Bas et al.'s method [34]	214.8089	221.1243
Yolcu et al.'s method [35]	201.0883	226.9612
Zhang et al.'s method [36]	52.9927	36.5687
TS-CNFS (proposed)	49.78939	33.64847

Note that each of the compared methods made the prediction of TAIEX or SSECI individually. In contrast, the proposed TS-CNFS performed the prediction of TAIEX and SSECI at the same time.

Experiment 2. The prediction of SSECI, S&P500 and Dow Jones is carried out by the proposed approach. All data are gathered from year 2006 to year 2012, where most of them are for training and the rest for testing. By the feature selection, thirty candidate features for each target are gathered, and there are total ninety candidate features., Three features eventually are selected from the ninety candidate features. The predictive model with two complex outputs for prediction of SSECI, S&P500 and Dow Jones. Performance comparison by the proposed model after training to the compared methods in literature is given in Table 4, showing the performance of the proposed model is superior to the compared methods, in terms of in terms of root mean square error.

Conclusion

The prediction system using complex fuzzy sets has been successfully developed and tested. The hybrid machine learning MCACO-RLSE method associated with the two-stage feature selection for the proposed model has achieved multi-target prediction in the experimentation. Through performance comparison with other methods in literature, as shown in Tables 3 & 4, the proposed approach has shown excellent performance.

Table 4: Performance Comparison in Root Mean Square Error (Experiment 2).

Method	SSECI	S&P 500	Dow Jones
BPNN [39]	30.8244	28.1231	286.6511
STNN [39]	29.0678	25.5039	258.3063
PCA-BPNN [39]	28.6826	20.5378	250.4738
PCA-STNN [39]	28.2975	19.2467	220.4365
SVM [39]	34.5075	25.9961	302.793
TS-CNFS (proposed)	27.61115	4.711123	21.41229

Note that each of the compared methods made the prediction of SSECI, S&P 500 or Dow Jones individually. In contrast, the proposed TS-CNFS performed the prediction of SSECI, S&P 500 and Dow Jones simultaneously.

Acknowledgement

This work was supported by the research projects MOST 105-2221-E-008-091 and MOST 104-2221-E008-116, Ministry of Science and Technology, TAIWAN.

References

1. Engle RF (1982) Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation Econometrica 50(4): 987-1007.
2. Bollerslev T (1986) Generalized Autoregressive Conditional Heteroscedasticity Journal of Econometrics 31: 307-327.
3. Widrow B, Hoff ME (1960) Adaptive Switching circuits Proc. IRE WESCON Convention Record. USA.
4. Kimoto T, Asakawa K, Yoda M, Takeoka M (1990) Stock market prediction system with modular neural networks Proc. International Joint Conference. USA.
5. Kim K, Han I (2000) Genetic algorithms approach to feature discretization in artificial neural networks for prediction of stock index Expert System with Applications 19: 125–132.
6. Roh TH (2007) Forecasting the volatility of stock price index Expert Systems with Applications 33: 916–922.
7. Zadeh LA (1965) "Fuzzy sets", Information and Control 8(3): 338-353.
8. Ramot D, Milo R, Friedman M, Kandel A (2002) Complex fuzzy sets IEEE Transactions on Fuzzy Systems 10(2):171-186.
9. Takagi T, Sugeno M (1985) Fuzzy identification of systems and its applications to modeling and control IEEE Trans. Syst. Man Cybern 15: 116–132.
10. C Li, Chiang T (2013) Complex Neuro fuzzy ARIMA Forecasting - A New Approach Using Complex Fuzzy Sets IEEE Transactions on Fuzzy Systems 21(3): 567-584.
11. Colorni A, Dorigo M, Maniezzo V (1991) Distributed optimization by ant colonies Proc. of the 1st European Conference on Artificial Life. France.
12. Socha K, Dorigo M (2008) Ant colony optimization for continuous domains European Journal of Operational Research 185(3): 1155–1173.
13. Shannon CE, Weaver W (1949) The Mathematical Theory of Communication Univ. of Illinois Press. USA.
14. E Parzen (1962) On estimation of a probability density function and mode Annals of Math. Statistics (vol. 33): 1065-1076.
15. Kwak N, Choi CH (2002) Input feature selection by mutual information based on Parzen window IEEE Transactions on Pattern Analysis and Machine Intelligence 24(12): 1667-1671.
16. Peng H, Long F, Ding C (2005) Feature selection based on mutual information: Criteria of maxdependency max-relevance and min-redundancy IEEE Transactions on Pattern Analysis and Machine Intelligence 27(8): 1226–1238.
17. Laney D (2001) 3D Data Management: Controlling Data Volume, Velocity and Variety META Group Research Note.
18. Juang C, Jeng T, Chang Y (2015) An Interpretable Fuzzy System Learned Through Online Rule Generation and Multiobjective ACO With a Mobile Robot Control Application. IEEE Transactions on Cybernetics. 46(12): 2706-2718.
19. Sadaei HJ, Enayatifar R, Lee MH, Mahmud M (2016) A hybrid model based on differential fuzzy logic relationships and imperialist competitive algorithm for stock price forecasting Applied Soft Computing 40: 132-149.
20. Kennedy J, Eberhart RC (1995) Particle swarm optimization in Proc. IEEE International Conference on Neural Networks (vol. 4). Australia.
21. Eberhart RC, Kennedy J (1995) A new optimizer using particle swarm theory in Proc. IEEE International Symposium on Micro Machine and Human Science. Japan.
22. Powell WB (2011) Approximate Dynamic Programming: Solving the Curses of Dimensionality John Wiley & Sons, Inc. Hoboken New Jersey. USA.
23. Kira K, Rendell L (1992) A practical approach to feature selection Proc. of the Ninth International Conference on Machine Learning. Scotland.
24. H Yu J Oh, Han WS (2009) Efficient feature weighting methods for ranking Proc. of the 18th ACM Conference on Information and Knowledge Management, China.
25. Clausius R (1854) Ueber eine vera nderte Form des zweiten Hauptsatzes der mechanischen Wa rmetheorie Annalen der Physik und Chemie 93(12): 481–506.
26. Cantor G (1874) Ueber eine Eigenschaft des Inbegriffs aller reellen algebraischen Zahlen Journal fu r die reine und angewandte Mathematik 77: 258-262.

27. Russell B (1923) Vagueness Australasian Journal of Philosophy 1(2): 84-92.

28. Max B (1937) Vagueness: An exercise in logical analysis Philosophy of Science 4:427–455.

29. McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity The bulletin of mathematical biophysics 5(4): 115-133.

30. Widrow B, Hoff ME (1960) Adaptive Switching circuits Proc. IRE WESCON Convention Record. USA.

31. Rumelhart DE, McClelland JL (1986) Parallel distributed processing: explorations in the microstructure of cognition vol. 1: foundations. MIT Press Cambridge MA, USA.

32. Socha K, Dorigo M (2008) Ant colony optimization for continuous domains European Journal of Operational Research 185(3): 1155–1173.

33. Huarng K, Yu HK (2005) A Type 2 fuzzy time series model for stock index forecasting Physical A: Statistical Mechanics and its Applications 35: 445-462.

34. Cheng CH, Cheng GW, Wang JW(2008) Multi-attribute fuzzy time series method based on fuzzy clustering Expert Systems with Applications 34(2): 1235-1242.

35. Chen SM (2002) Forecasting Enrollments Based On High-order Fuzzy Time Series Cybernetics and Systems 33(1): 1-16.

36. Lee LW, Wang LH, Chen SM, Leu YH (2006) Handling forecasting problems based on twofactors high-order fuzzy time series IEEE Transactions on Fuzzy Systems 14(3): 468-477.

37. Egrioglu E, Aladag CH, Yoclu U, Uslu VR, Erillli NA (2011) Fuzzy time series forecasting method based on Gustafson–Kessel fuzzy clustering Expert Systems with Applications 38(8): 10355-10357.

38. Wang L, Liu X, Pedrycz W (2013) Effective intervals determined by information granules to improve forecasting in fuzzy time series Expert Systems with Applications 40(14): 5673-5679.

39. Bas E, Yoclu U, Egrioglu E, Aladag CH (2015) A Fuzzy Time Series Forecasting Method Based on Operation of Union and Feed Forward Artificial Neural Network American Journal of Intelligent Systems 5(3): 81-91.

40. Yoclu OC, Yoclu U, Egrioglu E, Aladag CH (2016) High order fuzzy time series forecasting method based on an intersection operation Applied Mathematical Modeling 40(19-20): 8750-8765.

41. Zhang W, Zhang S, Zhang S, Yu D, Huand NN (2017) A multi-factor and high-order stock forecast model based on Type-2 FTS using cuckoo search and self-adaptive harmony search Neurocomputing 240: 13-24.

42. Hassan MR, Nath B (2005) Stock Market Forecasting Using Hidden Markov Model: A New Approach Proc. Fifth International Conference on Intelligent Systems Design and Applications. Poland.

43. Hsieh TJ, Hsiao HF, Yeh WC (2011) Forecasting stock markets using wavelet transforms and recurrent neural networks: An integrated system based on artificial bee colony algorithm Applied Soft Computing 11(2): 2510-2525.

44. Wang J, Wang J (2015) Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks Neurocomputing 156: 68-78.

45. Huarng KH, Yu THK, Hsu YW (2007) A Multivariate Heuristic Model for Fuzzy Time-Series Forecasting IEEE Transactions on Systems Man and Cybernetics Part B 37(4): 836-846.

46. Chen SM, Tanuwijaya K (2011) Multivariate fuzzy forecasting based on fuzzy time series and automatic clustering techniques Expert Systems with Applications 38(8): 10594-10605.

47. Chen SM (1996) Forecasting enrollments based on fuzzy time series Fuzzy Sets and Systems 81(3): 311-319.

48. THK Yu, Huarng KH (2008) A bivariate fuzzy time series model to forecast the TAIEX Expert Systems with Applications 34(4): 2945-2952.

DOI: 10.32474/CTCSA.2019.01.000115