ENCFIS: An Exclusionary Neural Complex Fuzzy Inference System for Robust Regression Learning

Chuan Xue and M. Mahfouf

***Abstract*—Robust learning, an emerging research topic in recent years, is a promising branch of advanced artificial intelligence. Robust learning models target mainly noisy and rough datasets, predominantly in situations where noises and outliers are hard to remove. In this paper, the concept of robust learning is combined with complex fuzzy theory for the first time, proposing a novel neuro-fuzzy system ENCFIS with extensive adaptability to numerical regression problems, with or without noise. Simulation results indicate that such architecture has excellent performance on a dataset with massive (45%) label noises and on a distorted time series dataset (25% corrupted). Additionally, experimental results on a metallurgy dataset also show that the approximation performance of ENCFIS is not compromised for the increase in robustness, making it an ideal candidate for general industrial scenarios with weak noise but difficult data characteristics.**

***Index Terms*—Robust machine learning, Neuro-fuzzy system, Complex fuzzy inference system (CFIS), Numerical regression.**

I. Introduction

FOR a long time, numerical modelling overridingly relied on classic statistical models such as ordinary least squares (OLS) [1], or hypothetical models based on tenable assumptions, expert knowledge, and deductive reasoning. However, the reliability of these methods is often inadequate, and, in many cases, they can only be employed to describe certain phenomena qualitatively, which hinders any future attempts to perform regression tasks based on the information obtained. With the rise of machine learning, inductive statistical models represented by artificial neural networks (ANN) are rapidly replacing traditional methods as the dominant approach in the field. As technology continues to permeate in this quickly digitizing world where internet of thing (IOT), deep learning and big data technologies are massively and increasingly applied, machine learning based numerical modelling is now indispensable among incalculable amount of research and industrial sectors.

Among all applications of numerical modelling, regression specifically benefits from machine learning methods, as algorithms of this category are capable of autonomously and precisely acquiring information from raw data, which enables it to remarkably outperform traditional methods, at least from the aspect of accuracy. Not the least of which, solutions of this category usually do not require its users to master precise mathematical expressions of the object. Although the knowledge of the data perspective is necessary to select the appropriate model for the task and verify the credibility of the final output, it is often easier to acquire than mathematical representations, which significantly reduces the difficulty of modeling. It is noteworthy that, in the realm of numerical regression, due to higher requirements for real-time performance compared with most of the pattern recognition scenarios, fuzzy machine learning algorithms, represented by ANFIS (Adaptive Neuro-Fuzzy Inference System) [2] and WM (Wang-Mendel method) [3], are often preferred owing to their simplicity, rapid convergence, competitive accuracy, and interpretability. After the emergence of interval type-2 fuzzy models [4], thanks to richer semantic implications of type-2 fuzzy logic, the prediction performance of the fuzzy algorithms is even further upgraded to a level that significantly outperforms most non-fuzzy machine learning methods.

Nevertheless, conventional machine learning methods including fuzzy logic based methods rely highly on the tidiness of the dataset. As such algorithms are all designed under the hypothetical premise that the distribution of data is clean, even the most preeminent algorithms can be confused if there are statistically notable noises or outliers in the data. Unfortunately, real-world data is rarely ideal. In many cases, it is feasible to remove the noise and outliers with certain preprocessing methods, such as filtering, smoothing or anomaly detection [5]. However, these methods are hardly 100% effective and anomalous data points will inevitably be fed into the model, leading to a reduction in performance. Some designs with good generalization capability may be resistant to such perturbations within data to a certain degree, but as long as such disturbances play a role in the final result, the regression accuracy of the model will more or less be affected. Even worse, if undetectable but destructive adversarial perturbation [6] is planted in the data, it will lead to major impact on model performance. In fact, practical attack methods [7] have already been developed by creating malicious data samples with specially designed adversarial perturbations for the purpose of sabotaging machine learning models. Conventional machine learning models are completely defenseless against this type of attack. As a matter of course, the performance and adaptability in real-world scenarios is likely to be enhanced if a machine learning regression model can initiatively counter outliers. This category of approaches is also referred to as "robust machine learning" [8].

Support vector regressions (SVRs) [9] can be considered as

This paper was produced by Department of Automatic Control and Systems Engineering, The University of Sheffield, Mappin Street, Sheffield S1 3JD (UK)

Manuscript received ?; revised ?.

a paradigm following this line of thought. From the perspective of statistics, the ultimate objective of all machine learning algorithms is to achieve the minimization of expected risk. In general, since the expected risk is difficult to measure, most algorithms seek to minimize the empirical risk, i.e., minimizing the mean loss over all samples in the training set. However, this poses a few more problems; pure empirical risk minimization tends to lead to over-fitting of the model and causes it to be sensitive to the noise. The concept of structural risk minimization is introduced in SVR as a compromise between empirical risk and expected risk, which can be captured by adding a L2 regularization in its loss function. Further, by introducing a slack variable to this regularization term, a "hard/soft-margin" is created, which pardons the errors arising from incorrect data points and thus significantly avoids the effects of noises [10]. Despite the fact that SVR has been used extensively to process noisy data, such method is not perfect. Firstly, although SVR can avoid the noise in many cases by setting a "margin", it does not recognize the differences between the noise and the normal data from the statistical aspect, which may cause the false adoption of noisy data samples while some normal ones are mistakenly excluded. Secondly, even a sparse representation is available for the SVR model, the complexity of training such a model can be excessive especially for nonlinear problems with a significant number of training samples, as it requires initializing a remarkable number of support vectors to obtain linearly separable results on the Hilbert space and calculate kernel functions for each support vector, which results in its exceptional inefficiency under such circumstances. In the worst case, if sparse representations are difficult to obtain, the number of support vectors for the SVR model may even approach the number of training samples, which is unacceptable.

Other attempts have also been made to improve the noise immunity of machine learning algorithms. The simplest way is to replace the objective function with a robust loss function, such as Huber loss [11], log-cosh loss or quantile loss [12]. Barron [13] proposed a generalized adaptive robust loss function, which is considered superior to traditional robust loss functions. This type of approaches has the advantage of simplicity, but limited improvement in algorithm robustness. Further, some researchers enhanced robustness by introducing a noise adaptation mechanism into the model, i.e., label-noise representation learning (LNRL) [14]. Goldberger *et al.* [15] increased the robustness of the algorithm to outliers by adding a noise adaptation layer to the deep network. Patrini *et al.* [16] introduced loss correction methods in deep learning models to form a more robustness model. Wang *et al.* [17] and Ren *et al.* [18] applied data reweighting techniques to facilitate the noise tolerance of network structures. Jiang *et al.* [19] employed a small-loss technique to suppress noise but resulted in cumulative errors. Han *et al.* [20] ameliorated the small-loss trick by training two neural networks synchronously to create a difference channel structure, which successfully cancels out cumulative noise. However, although the robustness of such models has been enhanced by LNRL methods, such train of thought also leads to a dramatic increase in training difficulty. Subsequently, Wang *et al.* [21] compensated for the shortcomings of previous LNRL methods by employing a meta-learning approach to estimate the noise transition matrix, which significantly ameliorates the efficiency of robust learning systems. This attempt also created a brand-new class of meta-learning based robust learning methods. Carmon *et al.* [22] proposed a semi-supervised learning strategy to allow networks to increase robustness by simply adding more unlabeled data samples, and theoretically justified this approach. Nevertheless, semi-supervised learning and meta-learning both require pure validation sets to pre-train the model, whereas for some scenarios clean data may not be available. In addition, there are some non-architectural level techniques that can improve robustness of the training process, including regularization, loss designing or even manually exclude some corrupted parts of data. Such methods are not strictly enhancements to the robustness of the model, but merely improve the robustness of the model for specific problems and therefore are not presented here. Despite the fact that many exploratory attempts have been made, conspicuous shortcomings exist in the present robust machine learning methods: a) The robustness comes at the expense of significantly offsets the accuracy of the algorithm. b) The training process is often sophisticated and unfriendly to applications requiring real-time adjustments. c) Mostly tailor-made to classification and pattern recognition tasks with uncertain effectiveness over numerical regression application scenarios. Notably, to our best knowledge, robust learning frameworks for fuzzy models have not yet emerged.

Driven by the above motivations, an exclusionary neural complex fuzzy inference system (ENCFIS) that learns from both normal and contaminated datasets with a robust, accurate and expeditious regression performance, has been proposed in this paper. Note that the word "exclusionary" in this context characterizes the model capability of ruling out the obfuscation caused by outliers during training. Complex fuzzy sets and logic (CFS&T) [23] is also introduced in this work to replace the traditional type-1 or type-2 fuzzy logic for constructing the fuzzy inference system. The membership degree of a complex fuzzy set is defined in a unit circle on the complex plane, with two-dimensional properties that enable less sensitivity to disturbance compared to the one-dimensional type-1 logic, contributing to the improved robustness. Additionally, complex numbers are algebraically closed structures, and the first-order derivatives of the membership functions of complex fuzzy sets can guarantee a closed-form solution, which makes gradient optimization possible. This is an overt advantage in contrast to the interval type-2 fuzzy logic because an algebraic solution to the derivative of an interval type-2 membership function does not exist. Considering that derivatives are actually the prior knowledge of the optimization surface, by applying gradient optimization policy, the complex fuzzy model can be potentially more efficient and precise than a typical interval type-2 model.

Regarding robust learning, the partition-based bisecting k-means clustering method [24] is employed to pre-train the antecedent parameters of membership functions, which not only ensures that the model is placed close to its optimum before the iterative optimization process but also helps to avoid the interference from label noises as clustering is performed only on input variables. Once the pre-trained antecedent membership function maps the training data into the Hilbert space, the M-estimator based on the Welsch influence function [25] performs noise-resistant robust estimation to determine the linear consequent parameters. The M-estimator can score the credibility of the training samples. It assigns low weights to data samples close to the noise and outliers in distribution and gives high weights to those without significant abnormalities to achieve parameter optimization in high-noise environments. Given that the residuals are often decoupled from the actual target under strong noise conditions, compromising the credibility of ordinary loss functions, a Huber loss function that can generate pseudo-residuals, is employed as the loss function, which significantly improves the reliability of the iterative training in high noise environments. Thus, ENCFIS is then able to perform iterative optimization to refine the network parameters for better performance. Finally, this robust design was tested on a highly contaminated synthetic dataset, a severely corrupted Sunspot time series dataset and a metallurgical dataset that simulates the real-world scenario. Experimental results indicate that the ENCFIS model is not just robust to extremely noisy datasets, but also hugely adaptable and competitive when it comes to practical applications.

The remainder of this paper is organized as follows: in Section II, related work on complex fuzzy theory, complex fuzzy systems and existing robust estimators is introduced. In Section III, the methodology of the ENCFIS, including network architecture, robust regression method, and optimization policy, is presented. In Section IV, the experimental implementations and outcomes are exhibited and discussed. Finally, conclusion as well as the discussions of future work will be given in Section V.

II. Related Work

A. Complex Fuzzy Theory & Systems

Ramot *et al.* [26], [27] proposed the pioneering complex fuzzy set & logic (CFS&L) by extending the common domain of the type-1 fuzzy membership from the interval between 0 and 1 to the entire unit circle in the complex plane. Sparked by the particle-wave duality in quantum physics, the modulus of the complex number is defined as the absolute value of the complex membership, corresponds to “particle”; while the phase is designated as the “context”, analog to “wave”. Thus, the complex fuzzy membership degree is denoted as follows:

$$\phi_s(x) = \varphi_s(x) \cdot e^{(j\omega_s(x))}, j = \sqrt{-1} \tag{1}$$

where $\varphi_S(x) \in [0,1]$ represents the modulus term and $\omega_S(x)$, which can be arbitrary value, signifies the phase term. If the phase term is set to be zero, then a complex fuzzy set will be reduced to ordinary type-1 fuzzy set. The fuzzy inference logic for complex fuzzy sets is also proposed in the work of Romat’s. Define two complex fuzzy sets A and B in the universe of discourse U, where set B is the aggregation of a group of sets $A_1, A_2, A_3, \ldots, A_n$. The operators for complement, union, intersection, and aggregation are defined respectively as follows:

$$\phi_{\bar{A}}(x) = C\left[\varphi_A(x)\right] \cdot e^{(j\omega_{\bar{A}}(x))} \tag{2}$$

$$\phi_{A\cup B}(x) = [\varphi_A(x) \oplus \varphi_B(x)] \cdot e^{(j\omega_{A\cup B}(x))} \tag{3}$$

$$\phi_{A\cap B}(x) = [\varphi_A(x) \star \varphi_B(x)] \cdot e^{(j\omega_{A\cap B}(x))} \tag{4}$$

$$\begin{aligned}\phi_A(x) &= aggregate[\phi_{A_1}(x), \phi_{A_2}(x), \ldots, \phi_{A_n}(x)] \\ &= \sum_{i=1}^{n} \omega_i \varphi_{A_i}(x),\end{aligned} \tag{5}$$

where $\oplus$ denotes t-conorm, and $\star$ is the operator for t-norm. In fuzzy mathematics, a t-norm refers to intersection in a lattice and conjunction in logic, whereas a t-conorm plays the role of a disjunction logic or a union operator. Dick [28] implemented a further investigation into Ramot’s work and developed a more general interpretation for complex fuzzy theory by introducing the mathematical concept of group theory. According to his point of view, the complex fuzzy sets can be divided into two factions, depending on whether the set is rotationally invariant. Rotational invariance is a notion of algebraic discipline. A function $L : \Gamma \times \Gamma \to \Gamma$ is defined as rotationally invariant if and only if $L\left(pe^{j\alpha} \cdot e^{jk}, qe^{j\beta} \cdot e^{jk}\right) = e^{jk} \cdot L\left(pe^{j\alpha}, qe^{j\beta}\right)$ holds for all $pe^{j\alpha}, qe^{j\beta} \in \Gamma$, where Γ is the lattice this set subordinates to. In abstract algebra, rotational invariance is also used to describe a group or a physical phenomenon. If a complex fuzzy set is considered as rotationally invariant, then the algebraic product cannot be applied for its conjunction operations. In accordance with the definition, Romat’s complex fuzzy theory belongs to the family of rotational invariance.

Dick’s discussion on the rotational invariance provided a theoretical foundation for the construction of complex fuzzy inference systems. Subsequently, fuzzy systems based on this novel theory have emerged. Man *et al.* [29] proposed the first complex neuro-fuzzy system by combining complex fuzzy logic with a single-input ANFIS and named it as Adaptive Neuro Complex Fuzzy Inference System (ANCFIS). Note that this six-layer system is mainly aimed at the prediction of quasi-periodic problems. Therefore, a sinusoidal membership function is created to attain this purpose:

$$r(\theta) = d\sin(a\theta + b) + c, \tag{6}$$

where a, b, c, d are premise parameters. However, the limitation of this network is obvious, as it can only tackle problems with periodic regularity, such as time series prediction. For broader function approximation problems, the scale of application is greatly restrained because of the shortcomings of the sinusoidal membership function.

To fill in this gap, Li *et al.* [30] proposed the complex neural-fuzzy system (CNFS) based on their originally developed polynomial complex Gaussian membership function, which is shown as follows:

$$\begin{aligned}\mu Gaussian(x, m, \sigma) &= \exp\left[-0.5(\frac{x-m}{\sigma})^2\right] \\ &-j(\frac{x-m}{\sigma^2}) \exp\left[-0.5(\frac{x-m}{\sigma})^2\right],\end{aligned} \tag{7}$$

where x is the base variable, m and σ denote the mean and the spread of a normal distribution. Compared with the sinusoidal function in ANCFIS, this Gaussian complex membership function can adapt to more universal purposes instead of only serving problems with periodicity. Such structure shows advantages in the application of a variety of function approximation problems. Nevertheless, the CNFS network is optimized using particle swarm optimization (PSO), which often gives rise to randomness and lack of robustness in training process. In addition, since the PSO algorithm does not utilize the first-order derivative, which is the prior knowledge of the optimization surface, its computational complexity is often a magnitude higher than that of gradient methods. Furthermore, due to the random initialization of the antecedent parameters, the training results heavily depend on initial values, which often leads to a local optimum. Clearly, there is still a lot of room for improvement of this structure.

Shoorangiz *et al.* [31] also proposed an adaptive complex neuro-fuzzy inferential system (ACNFIS) to target function approximation purposes. They applied a polar form complex Gaussian membership that directly inspired by the intersection operator from Ramot's complex fuzzy theory. This function is given as follows:

$$\mu(\theta)=\exp\left[-\left(\frac{\theta-c_A}{a_A}\right)^2\right]\angle\left\{2\pi\exp\left[-\left(\frac{\theta-c_P}{a_P}\right)^2\right]\right\} \tag{8}$$

where $\{c_A, a_A\}$ and $\{c_P, a_P\}$ are parameter sets for amplitude term and phase term respectively. Unfortunately, this study has not been tested in practice, probably due to the lack of simplicity as well as effective defuzzification methods for this polar form function compared to the previous two cases.

It is worth noting that the semantic interpretation of the complex fuzzy logic has not yet theoretically explained. Therefore, emerged complex fuzzy inference systems are all adaptive systems based on neural networks, which saves the trouble of understanding the complicated semantics. In fact, in addition to Ramot's complex fuzzy theory, other complex fuzzy theories have also been proposed. Consider that those works have not been used to construct fuzzy inference systems, thus will not be discussed here. Moreover, there is currently no research in the literature that applies complex fuzzy logic to build robust learning structures.

B. Robust Estimators

In the field of parameter estimation, many simple yet excellent methods are available, such as the least squares estimation, method of moments and maximum likelihood estimation [1], if the actual problem is consistent with the assumed parametric model and no statistically significant outliers exist in the observed values. However, in practice, idealized parametric models are rarely identical to real problems, and observations are often inaccurate, noisy, or distorted. In this context, the concept of robust statistics [32] came into being, which is characterized for its tolerance to imprecision in hypothetical models as well as to the interference of abnormal observations. Robust estimators, designed based on the idea of robust statistics, strive to approach traditional statistical methods but are more resistant against misleading outliers or deviations from the reference statistical distribution.

A variety of robust estimators have indeed emerged during the past few decades, which can be classified into three main categories, namely L-estimators, R-estimators, and M-estimators. Specifically, L-estimation is a special linear combination of statistics of the observation, for example, the sample median and α-trimmed mean, which is extraordinarily straightforward but statistically robust. The most famous case of L-estimation is the Theil-Sen estimator [33], [34], in which the mean of a least square estimator is replaced by the multivariate median, hence significantly increases the robustness. Such method has no additional parameters, and its performance is even comparable to OLS, with a high breakdown point of 29.3%. However, due to the difficulty of determining the median value increases sharply with the raise of dimension, the efficiency and accuracy of the L-estimator represented by Theil-Sen estimator plummets under high-dimensional conditions. Consequently, L-estimators are rarely taken in practice since high-dimensional scenarios are often involved in linear estimations. L-estimators based on α-trimmed mean are also not preferred because of their dependance on prior knowledge. Differ from L-estimation, R-estimation is realized through the classification of residues, in which the residuals are compartmentalized into several classes and each class is assigned a unique weight, therefore achieving a robust statistic. However, the R-estimator is used even less than L-estimators because its construction even more relies on the prior knowledge of the problem, which vastly increases the arduousness of its applications.

Currently, the most mainstream robust estimation method is the M-estimation. The difference between this method and the previous two is that it simulates the possible distribution of the data by formulating an influence function, thus avoiding the requirement for prior knowledge. The M-estimator, as a generalization of the maximum likelihood estimator, is susceptible to the data distributed near the mean while insensitive to the disturbance of anomalous points (usually situated far from the mean). Therefore, a reliable and robust estimation can be accomplished even if the priori of the data is unavailable. In general, there are two types of M-estimators, i.e., ρ-estimator and ψ-estimator. For ρ-type, assume an n-dimensional measure space $\Lambda \in \mathbb{R}^n$, and $\lambda \in \Lambda$ is the parameter vector of the model. Thus, the representation of this Mestimator $\Xi(G)$ in accordance with the mapping $\rho : \chi \times \Lambda \to \mathbb{R}^n$ is given as follows:

$$\Xi(G) := \operatorname{argmin}_{\lambda\in\Lambda} \int_\chi \rho(x, \lambda) dG(x), \tag{9}$$

where $\rho(x, \lambda)$ is the influence function, G denotes the distribution of observed values and χ refers to the distribution of estimation. Note that if $\rho(x, \lambda) = -\ln\left(\frac{\partial G(x,\lambda)}{\partial x}\right)$, then the M-estimator will reduce to the ordinary maximum likelihood estimator. For a differentiable and continuous influence function ρ, the M-estimator can be simplified to a more computationally convenient form, i.e., the ψ-type. In this case, the estimator $\xi(G)$ is defined by equation:

$$\int_\chi \psi(x, \xi(G)) dG(x) = 0, \tag{10}$$

and the estimation $\tilde{\lambda}$ can be obtained by solving the following equation:

$$\int_{\mathcal{X}} \psi(x,\lambda)dG(x) = 0 \tag{11}$$

where $\psi(x,\lambda) = \frac{\partial\rho(x,\lambda)}{\partial\lambda}$. In addition, M-estimators with bounded ψ are typically robust, which suggests the importance of selecting appropriate influence functions. Dozens of influence functions have been proposed so far, though, only five of those are most used, i.e., the Huber function, the Hampel function, the Cauchy function, the Bisquare function and the Welsch function [25].

III. Methodology of The Proposed Method

In this part, network architecture, optimization policy, and robust learning strategies for the proposed ENCFIS algorithm will be investigated. To ensure accuracy, the Sugeno defuzzification [35], which is considered as the most exactitude, is employed. Assume the objective system has m inputs and one output, then the i th fuzzy rule can be represented as follows:

Rule i : IF l_1 is $A_1^i(x_1)$ and l_2 is $A_2^i(x_2), \dots,$ and l_m is $A_m^i(x_m)$

By enforcing the Sugeno defuzzification, the output ζ^i of this rule is obtained:

$$\zeta^i = a_0^i + \sum_{j=1}^{m} a_j^i x_j, i = 1, 2, \dots, n, \tag{12}$$

where $A_j^i(x_j)$ denotes the j th antecedent of the i th complex fuzzy rule, a_j^i signifies its corresponding consequent parameter. In terms of the optimization of the antecedent parameters, the bisecting k-means algorithm is utilized first to pre-train the Gaussian kernel parameters to approximate the actual global optimum. Subsequently, the DEMON [36] momentum decaying optimization is taken to further fine-tune the parameters. Regarding the consequent part, a ψ-type M-estimator based on the Welsch influence function is applied to conduct a robust linear estimation in order to determine the consequent parameters from noisy datasets. Moreover, Huber loss is selected as the cost function to counteract the disturbance caused by outliers. An overall schematic diagram of the ENCFIS is unfolded in Fig. 1.

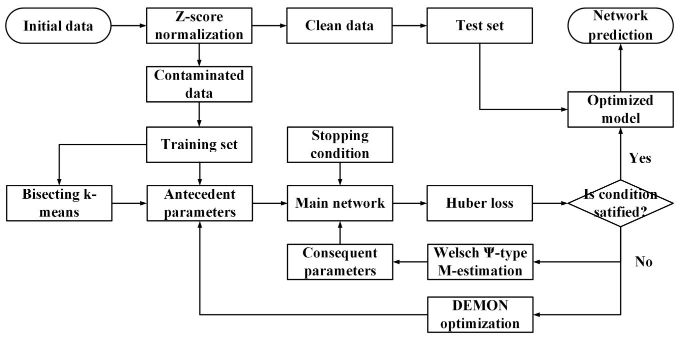

Fig. 1. ENCFIS algorithm flow.

A. Network Structure of ENCFIS

ENCFIS has a five-layer network architecture which is displayed in Fig.2. A complex Gaussian membership function in polynomial form similar to the one in CNFS [30] is employed. This is a batch learning architecture similar to the initial version of ANFIS [2], where the parameters are updated after every complete iteration. The operation of each layer under the m-dimensional input vector $x(\tau) = [x_1(\tau), x_2(\tau), \dots, x_m(\tau)]^T$ at time τ is presented as follows:

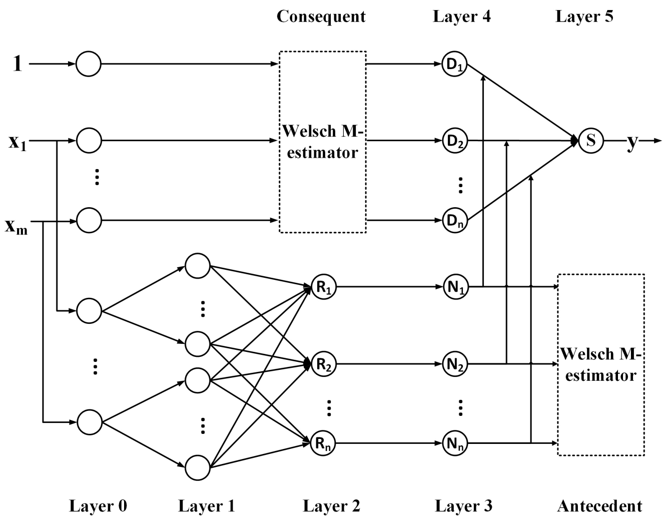

Fig. 2. The main network structure of ENCFIS. (X_i denotes input variable, R_i represents the firing strength of each rule, N_i is the normalization layer, D_i is the outcome of Sugeno defuzzification, and S signifies the sum of the previous layer, which is identical to the output of the network, i.e., y.)

Layer 1: The fuzzification is carried out in this layer, where the real-valued inputs are mapped into the complex common domain via the following complex Gaussian membership function:

$$O_{1,j}^i(\tau) = \exp\left(-\frac{\left(x(\tau)-\mu_j^i\right)^2}{2b_j^i}\right) - j\exp\left(-\frac{\left(x(\tau)-\mu_j^i\right)^2}{2b_j^i}\right)\frac{x(\tau)-\mu_j^i}{b_j^i}\lambda_j^i, \tag{13}$$

where $O_{1,j}^i(\tau)$ refers to the complex membership of the i th rule by the j th input variable, and $\{\mu_j^i, b_j^i, \lambda_j^i\}$ denotes the antecedent parameters for each node.

Layer 2: This layer computes the firing strength of each rule, in which a complex multiplication is applied:

$$O_2^i(\tau) = \prod_{j=1}^{m} O_{1,j}^i(\tau) =: \alpha_j^i(\tau) + j\beta_j^i(\tau) \tag{14}$$

$$\alpha_j^i(\tau) = \left[1 - \prod_{j=1}^{m} \frac{\left(x_j(\tau)-\mu_j^i\right)}{b_j^i}\lambda_j^i\right]\exp\left(-\sum_{j=1}^{m}\frac{\left(x_j(\tau)-\mu_j^i\right)^2}{2b_j^i}\right) \tag{15}$$

$$\beta_j^i(\tau) = -\sum_{j=1}^{m}\lambda_j^i\frac{x_j(\tau)-\mu_j^i}{b_j^i}\exp\left(-\sum_{j=1}^{m}\frac{\left(x_j(\tau)-\mu_j^i\right)^2}{2b_j^i}\right), \tag{16}$$

where $O_2^i(\tau)$ is the strength of the i th firing and $\alpha_j^i(\tau), \beta_j^i(\tau)$ represent the value of real component and imaginary component, respectively.

Layer 3: All firing strengths are normalized in this layer via complex division to thoroughly interact the two dimensions of information. For brevity, formulas (15) and (16) are used here to simplify the expression:

$$O_3^i(\tau)=\frac{O_2^i(\tau)}{\sum_{r=1}^n o_2^r(\tau)}=\frac{\alpha^i(\tau)+j\beta^i(\tau)}{\sum_{r=1}^n \alpha^r(\tau)+j\sum_{r=1}^n \beta^r(\tau)}, \quad (17)$$

where $O_3^i(\tau)$ denotes the normalized output of i th node in this layer.

Layer 4: Such layer consists of adaptive nodes, in which the Sugeno defuzzification is implemented to derive the output of fuzzy reasoning in accordance with corresponding consequent parameters:

$$O_4^i(\tau) = O_3^i(\tau) * \left(p_0^i + \sum_{j=1}^m p_j^i x_j(\tau)\right), \quad (18)$$

where $O_4^i(\tau)$ is the result of each reasoning and $\{p_0^i, p_1^i, p_2^i, \ldots, p_m^i\}$ are consequent parameters.

Layer 5: The overall output of the network is determined in this layer by simple sum of outputs from layer 4. Note that only the real component of the complex output is utilized.

$$O_5(\tau) = \mathrm{Re}\sum_{i=1}^n O_4^i(\tau), \quad (19)$$

where "Re" denotes the real component of the complex number, $O_5(\tau)$ is the network output and m equals to the number of fuzzy rules for this system. For a real-valued regression task, the expected result is real-valued, and one cannot assume the process is complete if the outcome is complex-valued instead. This operation of mapping the complex reasoning result into real numbers is essentially a step of defuzzification for the complex fuzzy sets. For a complex fuzzy membership function defined in polynomial form, its complex plane projection on the real axis equals to the real component, which makes it the best candidate to be the defuzzified output. Not the least of which, this strategy maintains the nature mapping relation between the fuzzy antecedents and the output, enabling the antecedent parameters to be pre-train by a partition-based clustering method. Technically, ENCFIS shares the same feed-forward structure as CNFS [30], but its learning method has shifted from the online learning of CNFS to a batch learning strategy to serve robust learning purposes.

Regarding the selection of objective function, the mean square error (MSE) loss function, is often preferred owing to its good performance in regression tasks. But MSE loss is very sensitive to outliers, causing a vulnerability against noisy data. Although the mean absolute error (MAE) is considered more robust against disturbances, it is rarely applied in neural network optimizations due to its constant gradient that performs poorly when approximating the minimum. Hence, Huber loss [11], combining the merits of the two, is taken as the objective function for ENCFIS, which is shown as follows:

$$\widehat{H}(\tau)=\begin{cases} \frac{1}{2}\left(y(\tau)-O_5(\tau)\right)^2, & |y(\tau)-O_5(\tau)| \le \delta \\ \delta\left|y(\tau)-O_5(\tau)\right|-\frac{1}{2}\delta^2, & otherwise \end{cases} \quad (20)$$

where $\widehat{H}(\tau)$ represents the Huber loss, $|y(\tau) - O_5(\tau)|$ is the residual at the τ th input, δ denotes the tuning coefficient. By setting a δ, the Huber function diverts a part of abnormal data points to the MAE loss, which enhances the robustness of the traditional MSE loss. After deploying the Huber loss, the optimization process no longer relies on the residual, but the pseudo-residual, which is defined as:

$$\hat{h}(\tau)=\begin{cases} y(\tau)-O_5(\tau), & |y(\tau)-O_5(\tau)| \le \delta \\ \delta sign\left(y(\tau)-O_5(\tau)\right), & otherwise \end{cases} \quad (21)$$

where $\hat{h}(\tau)$ represents the pseudo-residual generated by the τ th data point, $sign()$ is the signum operator that extracts sign from a real number. Since noise tends to keep the actual residuals away from the optimization target, the introduction of pseudoresiduals certainly smooths the training process.

B. Bisecting K-Means

K-means clustering is the best-known clustering method, simple but effective and are massively applied in many fields. The primitive k-means, as the name suggests, divides the data into k clusters according to the pre-determined centroids $\xi_1, \xi_2, \ldots, \xi_k$. By calculating the distance between each data point x_i and the cluster center ξ_j, each point will be distributed to the cluster that has the smallest point-to-center gap. Then, centroids are relocated in accordance with the new clusters:

$$\xi_j = \frac{1}{|q_i|}\sum_{x\in q_i} x \quad (22)$$

where q_i refers to the j th cluster, $j = 1, 2, \ldots, k$. Via multiple repetitions of above procedures, the optimized solution for the clusters is determined. However, this method is extremely dependent on the selection of the initial centroids. Whether it is the manual or random initialization, the result can easily fall into the local optimum.

To address this issue, Steinbach *et al.* [24] combined the notion of hierarchical clustering with the primeval k-means, creating a novel top-down version which is now known as the bisecting k-means. This algorithm also relies on stochastically chosen of initial centroids, but it effectually averts local optimum with the help of a binary decision tree. To implement the bisecting k-means, first consider the whole dataset as a single cluster, and ordinary k-means is performed to bisect it. The sum of the squared error (SSE) value for each cluster is then calculated, and the one with the larger SSE will be further halved. The above step will be repeated until the number of clusters equals to the preset. SSE is determined by the following formula:

$$SSE = \sum_{i=1}^n (x_i - \bar{x})^2 \quad (23)$$

where $\bar{x}$ is the mean of the cluster, x_i denotes an individual point and there are n points in this cluster in total. Compared with the earlier k-means counterparts, the bisecting k-means converges faster especially when the preset k is large, and the result can be guaranteed to be close to the global optimum.

C. DEMON Momentum Decaying Optimization

The gradient-momentum method [37] is widely adopted amongst the brigade of gradient descent schemes. The idea of momentum is introduced to facilitate optimization in directions of low curvature, while avoiding fluctuations in directions of high curvature. Compared with the vanilla gradient solution, the addition of momentum smooths the optimization process and helps avoid some possible local optima. Assume $x(k) \in \mathbb{R}^d$ is the parameter vector of the k th iteration for a network, the iterative expression for the classical gradient-momentum optimization is as follows

$$\left\{\begin{array}{l} x(k+1)=x(k)+\alpha v(k) \\ v(k)=\eta v(k-1)-\nabla f(x(k)), \end{array}\right. \tag{24}$$

where $\alpha \in \mathbb{R}$ represents the learning rate, $\nabla f(x(k))$ refers to the vector of the gradient, η denotes the momentum coefficient and the item $v(k) \in \mathbb{R}^d$ accumulates momentum over the iterations. However, such technique is very sensitive to the initial parameter setting, and even subtle changes can make a huge difference in the final optimization outcomes. In addition, unchanged momentum also causes it easy for the model to miss the optimum over the last few iterations. Consequently, methods of gradually reducing momentum during training were proposed, in which DEMON [36] is currently considered the state-of-the-art.

It is well-known that the concept of momentum is essentially the impact of historical gradients on the present learning process. For a constant momentum training process, the main iterative expression can be rewritten as:

$$\begin{aligned} x(k+1)= & x(k)-\alpha \nabla f(x(k))-\alpha \eta \nabla f(x(k-1))- \\ & \cdots-\alpha \eta^{k-1} \nabla f(x(1)) \\ = & x(k)-\alpha \nabla f(x(k))-\alpha \sum_{i=1}^k \eta(i) \nabla f(x(k-i)) \end{aligned} \tag{25}$$

where $\eta(i)$ signifies the momentum of the i th iteration and in this case $\eta(i)=\eta^i$. It should be noted that if $k \rightarrow \infty$, then $\eta(k) \rightarrow 0$. According to the sum of geometric series, we have:

$$\lim_{k \rightarrow \infty} \sum_{i=1}^k \eta(i)=\eta \lim_{k \rightarrow \infty} \sum_{i=0}^k \eta(i)=\frac{\eta}{1-\eta}. \tag{26}$$

The DEMON momentum decaying is developed enlightened by the above expression such that the momentum can be reduced to zero within finite T steps. In accordance with this idea, the decayed momentum at the k th step is given as:

$$\eta(k)=\eta_{init} \frac{1-\frac{k}{T}}{\left(1-\eta_{\text{init}}\right)+\eta_{init}\left(1-\frac{k}{T}\right)}, \tag{27}$$

where η_{init} is the initial momentum. Thus, the renewed gradient-momentum update equation applying DEMON momentum decay is as follows

$$\left\{\begin{array}{l} x(k+1)=x(k)-\alpha \nabla f(x(k))+\eta(k) v(k) \\ v(k)=\eta(k-1) v(k-1)-\alpha \nabla f(x(k-1)) \end{array}\right., \tag{28}$$

Since the momentum decays to zero after finite iterations, the sensitivity of the network to the initial value of the momentum parameter sharply decreases. This is a huge advantage for models seeking for more robustness during training, especially when a model is extremely hyperparameter sensitive. Such scheme also enables a better optimization process around the optimal point.

D. M-Estimator with Welsch Influence

Normally, the linear consequent part of a Sugeno neuro-fuzzy system is updated utilizing the ordinary LS. However, ordinary LS is susceptible to abnormality among the training samples, making it unqualified as part of a robust learning architecture. Thus, an M-Estimator with the Welsch influence is applied to estimate the linear parameters of the network. The Welsch influence function is defined as follows:

$$f_W(\varepsilon)=\frac{\gamma^2}{2}\left[1-\exp \left(-\frac{\varepsilon^2}{\gamma^2}\right)\right], |\varepsilon| \leq \infty \tag{29}$$

where the symbol γ denotes the tuning constant for an M-estimator and ε refers to the independent variable of the influence function. Given the following system model:

$$y_L=S_L \theta+\omega_L, \tag{30}$$

where S_L is the sample matrix, θ denotes the parameter vector, y_L is the vector for output labels, ω_L is the vector of the system noise which is also independent to the input. The discrete M-estimator can be obtained via minimizing the following objective function:

$$\sum_{i=1}^n f_W\left(\frac{y_i-s_i \hat{\theta}}{\hat{\sigma}}\right)=\sum_{i=1}^n f_W\left(\varepsilon_i\right), \tag{31}$$

where $s_i \in S_L$ denotes the i th data sample, $\hat{\sigma}$ represents the scale parameter, ε_i is the i th z-scored residual, f_W refers to Welsch influence. For continuously differentiable function f_W, designate $\psi=\frac{\partial f_W}{\partial \theta}$, then the minimization process can be further simplified as solving the following equation:

$$\sum_{i=1}^n \psi\left(\frac{y_i-s_i \hat{\theta}}{\hat{\sigma}}\right) s_i=\sum_{i=1}^n \psi\left(\varepsilon_i\right) s_i=0, \tag{32}$$

Define weight function $w(\varepsilon)$ under the Welsch influence:

$$w(\varepsilon)=\frac{\psi(\varepsilon)}{\varepsilon}=\exp \left(-\frac{(\varepsilon / \gamma)^2}{2}\right), \tag{33}$$

and substitute it into equation (32), we have:

$$\sum_{i=1}^n w\left(\varepsilon_i\right) \varepsilon_i s_i=0 \tag{34}$$

To estimate $\hat{\theta}$, the following equation is derived by substituting $\varepsilon_L=S_L \hat{\theta}-y_L$ into equation (34):

$$S_L^T W S_L \hat{\theta}-S_L^T W y_L=0, \tag{35}$$

where W is the weight matrix determined by the weight function $w(\varepsilon)$. Thus, the final estimation of the parameter vector θ is obtained as follows:

$$\hat{\theta}=\left(S_L^T W S_L\right)^{-1} S_L^T W y_L \tag{36}$$

Regarding the tuning constant γ in the weight function, this is related with the estimation efficiency. In applied statistics, the efficiency of ordinary LS is usually used as the reference for evaluating the efficiency level of a robust estimator [25]. The higher the constant tunes, the closer its performance is to LS, but the robustness against outliers decreases. The most encountered relative efficiencies of Welsch Mestimator against ordinary LS under different γ settings are given in TABLE I [25]. In most cases, the M-estimator is recommended to run at 95% efficiency level.

TABLE I
THE EFFICIENCY LEVEL UNDER DIFFERENT γ FOR WELSCH M-ESTIMATOR.

γ	2.3831	2.9850	3.9104	4.7407
Efficiency level	90%	95%	98%	99%

E. Robust Learning Process for ENCFIS

In this section, the robust learning method utilizing bisecting k-means, DEMON momentum decaying, Welsch M-estimator as well as Huber loss is illustrated. Note that if the variables of the input data are too different in scale, it may negatively affect the performance of clustering, gradient optimization, and M-estimator. Thus, an essential factor for this strategy to work is that all data should be normalized according to Z-score [38] formula:

$$x_z = \frac{x-\mu_z}{\sigma_z} \tag{37}$$

where μ_z is the mean of the samples, σ_z is the standard deviation, and x_z is the normalized value. The reason for using the Z-score instead of the Max-Min normalization here is that the data processed by ENCFIS usually suffer from a significant amount of outliers, and the scale of the outliers may be larger than the correct data samples. Z-score can unify the data of different scales and avoid the normalization outcome being swayed by the outliers. After the standardization, clustering is employed to pre-train the antecedent part of the network using only input variables. By operating the bisecting kmeans on all training samples, the vector of r cluster centroids $C = \{c_1, c_2, \ldots, c_i, \ldots, c_r\}$ is obtained. Here, r must be identical to the number of fuzzy rules that assigned to each input variable. Subsequently, the sum of the distances between samples and their corresponding centroid in every cluster is calculated, i.e., $D = \{d_1, d_2, \ldots, d_i, \ldots, d_r\}$, where $d_i = \sum_{k=1}^{n}\left(x_i^k - c_i\right), n$ denotes the number of samples in this cluster. Therefore, the antecedent parameter set $\{\mu^i, b^i, \lambda^i\}$ is determined as follows:

$$\mu^i = c_i, b^i = \rho\sum_{k=1}^{n}\left(x_i^k - c_i\right), \lambda^i = 1 \tag{38}$$

where μ^i denotes the center of each kernel, b^i refers to the width and λ^i represents the scale factor for the complex component. ρ is said to be the expansion factor, which is default to be 1.

For the conventional type-1 Gaussian fuzzy kernels, after the previous step, the antecedent parameters should be very close to the global optimum. But for the complex fuzzy antecedents, there is still a gap to fill in, as the clustering algorithm estimates only the projection of the complex membership functions on the real axis. The remaining part is left to the iterative optimization algorithm to approximate its actual condition in the complex plane. For ENCFIS, the DEMON gradient optimization strategy is utilized to refine the parameters. Combine the equations (27) and (28), the expression of the update rule for parameter μ^i at the recursion $k+1$ is given as follows:

$$\begin{cases} \eta(k) = \eta_{init}\frac{1-\frac{k}{T}}{(1-\eta_{\text{init}})+\eta_{\text{init}}\left(1-\frac{k}{T}\right)} \\ \mu^i(k+1) = \mu^i(k) - \alpha h(k)\left.\frac{\partial f(k)}{\partial \mu^i}\right|_{\mu^i=\mu^i(k)} + \eta(k)v(k) \\ v(k+1) = \eta(k)v(k) - \alpha h(k)\left.\frac{\partial f(k)}{\partial \mu^i}\right|_{\mu^i=\mu^i(k)} \end{cases} \tag{39}$$

where $h(k)$ is the pseudo-residual under the Huber loss, which is already defined by equation (21). $\frac{\partial f(k)}{\partial \mu^i}$ is the partial derivative with respect to μ^i. Further, the update rules for parameters b^i and δ^i can be obtained by replacing $\left.\frac{\partial f(k)}{\partial \mu^i}\right|_{\mu^i=\mu^i(k)}$ with $\left.\frac{\partial f(k)}{\partial b^i}\right|_{b^i=b^i(k)}$ and $\left.\frac{\partial f(k)}{\partial \lambda^i}\right|_{\lambda^i=\lambda^i(k)}$, respectively.

As for the linear consequent parameters, the Welsch M-estimator is utilized to robustly estimate the optimum. Assume there is a set of N data samples involved, then define the equation coefficient matrix S^i, weight matrix W^i, data label vector y and the parameter vector θ^i as follows:

$$S^i = \begin{bmatrix} f_0^i(x_1) & f_1^i(x_1) & \cdots & f_q^i(x_1) \\ f_0^i(x_2) & f_1^i(x_2) & \cdots & f_q^i(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_0^i(x_N) & f_1^i(x_N) & \cdots & f_q^i(x_N) \end{bmatrix},$$

$$W^i = \begin{bmatrix} w_1^i & \cdots & 0 & 0 \\ 0 & w_2^i & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_N^i \end{bmatrix},$$

$$y = [y_1, y_2, \ldots, y_N]^T,$$

$$\theta^i = \left[a_0^i, a_1^i, \ldots, a_q^i\right]^T,$$

where $f_0^i, f_1^i, \ldots, f_q^i$ denote the outputs of the antecedent neurons, q equals to the number of fuzzy rules, and $w_0^i, w_1^i, \ldots, w_N^i$ are determined in accordance with formula (33). Hence, the estimated consequent parameter at the k th recursion can be calculated as:

$$\hat{\theta}^k = \left[\left(S^k\right)^T W^k S^k\right]^{-1}\left(S^k\right)^T W^k y. \tag{40}$$

To correctly implement the proposed robust learning process, the clustering step only needs to be used once, whereas the DEMON optimizer and M-estimator joint learning approach is supposed to repeat several recursions until the error requirement is satisfied or the maximum epoch setting is reached. Such learning process is extremely robust against the label noise for the following reasons. Firstly, the clustering process does not involve labels, thereby more accurately estimates the antecedent parameters. Secondly, the real residuals are replaced with pseudo residuals to conduct gradient optimization, which offsets the interference of noise. Thirdly, the noise-insensitive M estimator is adopted to obtain the consequent parameters, which avoids the problems caused by ordinary LS. Note that the choice of gradient optimization strategy needs to be a smooth and hyperparameter-insensitive gradient-momentum solution similar to Demon. In early trials, for clean datasets, most gradient optimization methods achieved close outcomes on ENCFIS, but for highly noisy data, smooth gradient methods have a better chance to converge because a solution that relies on the stochastic gradient for the global optimum, such as SGD or Adam, would fail to converge due to the misleading residuals caused by a large volume of noise. Also, hyperparameter-sensitive solutions may significantly increase the difficulty of determining the parameters due to the noise and are therefore not recommended. The simplified algorithm procedures are given as follows:

Algorithm 1 Robust Learning for ENCFIS (proposed).

```
Step 1. Normalize the dataset according to Z-score standardization formula.
Step. 2 Pre-determine the antecedent parameter set $\{\mu^i, b^i, \lambda^i\}$ using information obtained from bisecting k-means according to equation (38).
Step. 3 Compute outputs with the initial setting, then generate the pseudo-residual $\hat{h}(\tau)$ under the Huber loss.
Step. 4 Refine the antecedent part (non-linear) of the network by DEMON momentum decaying method.
Step. 5 Estimate the consequent part (linear) with Welsch M-estimator.
Step. 6 Calculate the output of the network, obtain the renewed pseudo-residual $\hat{h}(\tau)$.
Step. 7 Repeat Step.4-Step.6 until the stopping condition is satisfied. (It is recommended to set the maximum number of iterations as the stopping condition.)
```

IV. Experiments and Discussions

A total of three experiments are included in this section to test the performance of the proposed robust learning architecture. It should be noted that all simulations in this paper were implemented on MATLAB 2022b environment, and the benchmark models run for the tests were mainly from MATLAB's built-in toolbox, while the rest were programmed from scratch.

Additionally, four different indicators are involved in this part to evaluate the performance, i.e., standard deviation (STD), root-mean-square deviation (RMSE), mean absolute error (MAE), and symmetric mean absolute percentage error (SMAPE). The calculation formulas are given as follows:

$$STD = \left(\frac{1}{N-1}\sum_{i=1}^{N}(y_i - u)^2\right)^{\frac{1}{2}} \tag{41}$$

$$RMSE = \left(\frac{1}{N}\sum_{i=1}^{N}(y_i - w_i)^2\right)^{\frac{1}{2}} \tag{42}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - w_i| \tag{43}$$

$$SMAPE = \frac{100\%}{N}\sum_{i=1}^{N}\frac{2 * |y_i - w_i|}{|y_i| + |w_i|}, \tag{44}$$

where N denotes the number of the samples; u is the mean of the samples; y_i is the label and w_i is the estimated value; $y_i - w_i$ represents the residual.

A. *Corrupted Synthetic Data Test*

The single-input-single-output synthetic dataset is produced by the standard sinusoidal function:

$$f(x) = \sin(x-3), 0 \leq x \leq 10, \tag{45}$$

in which 201 original data points are isometrically sampled from [0,10] with an interval of 0.05. Corruption is then added to the data, by evenly replacing initial data points with random points ranged from [-3, 2]. A total of 45% of the data points were "corrupted" during this process, which poses a huge challenge to any machine learning model. It is worth noting that there is no testing or validation set for this experiment. Algorithm performance is evaluated based on its ability to "restore" the original data. However, given that the RMSE computed using corrupted data makes no sense, RMSE value will be evaluated according to the original data. The hyperparameter setting for this test is given in TABLE II and the comparison between the proposed ENCFIS model and other models is revealed in TABLE III. The actual MSE as well as the Pseudo-MSE during the training process are side by side displayed in Fig. 3. The visualization of the data restored by different models is shown in Fig. 4.

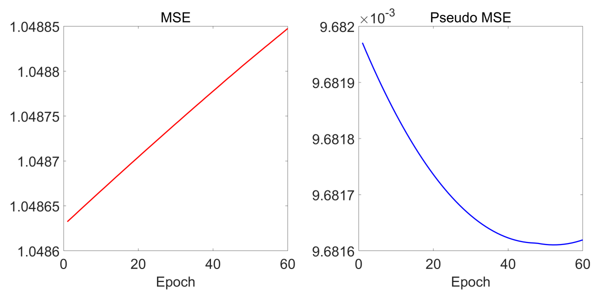

Fig. 3. MSE and Pseudo-MSE in training.

TABLE II
ENCFIS hyperparameter setting for the corrupted synthetic data test.

Bisecting k-means clustering for antecedence pre-training			
Cluster number k	9	Expansion factor ρ	1
DEMON		M-estimator and Huber Loss	
Learning rate α	10^{-6}	Tuning constant γ	2.9850
Initial momentum factor β	0.1	α-cut factor δ	0.15

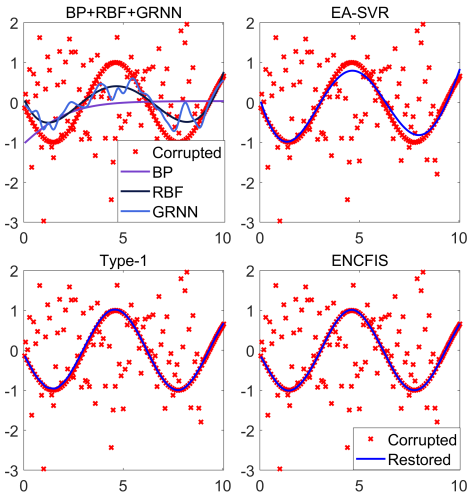

Fig. 4. Restored sinusoidal curves.

As it can be easily seen from Fig.3 that the variation trends of MSE and pseudo-MSE values during training, computed in accordance with residuals and pseudo-residuals respectively, show a big difference. The actual residuals from the training process have lost their authenticity as a measure of the empirical error, due to the very large number of noisy data points in the dataset. Despite that, pseudo-residuals generated by the Huber loss function have played a crucial role in measuring the training error, so that the training can proceed in the right direction. Regarding the "restoration" performance, six models including BP [39], RBF [40], GRNN [41] , EA-SVR, Type-1 and ENCFIS are involved in this test, where Type-1 is a counterpart of ENCFIS that uses exactly the same robust learning strategy but takes the traditional type-1 fuzzy logic instead of the complex fuzzy logic. Type-1 model is specially introduced as a control group to find out if the complex fuzzy logic is conducive to performance and robustness of the model. Furthermore, EA-SVR is a SVR model that applies evolutionary algorithm [42] to tune the hyperparameters, for obtaining the best performance from the SVR model.

As shown in Fig. 4, the mostly used but non-robust regression models such as BP, RBF and GRNN are completely ineffective under such a high level of noise. Even the SVR, the most classical de-facto robust regression learning method, which is considered very resistant to noise, has been influenced, despite the adoption of EA to tune its parameters. This also confirms that SVR's strategy of offsetting noises via setting a "margin" has limitations. However, the proposed ENCFIS model as well as the "Type-1" model obtained the best outcome in this data recovery challenge. Further investigation of the RMSEs in TABLE III reveals that ENCFIS not only far outperforms all models including SVR, but also significantly outperforms Type-1, which is the type-1 fuzzy logic counterpart of ENCFIS as mentioned above. Note that the RMSE values here were obtained by comparing the restored data with the uncorrupted original data. Given that experimental results may suffer from coincidence, the Kruskal-Wallis test [43] is introduced to ensure that the conclusion holds from the statistical perspective. To carry out the significance test, EA-SVR, Type-1, and ENCFIS are executed 20 times on this dataset to obtain a set of RMSE values for each model as the performance reference. Then the narrative "the two sets of RMSE values have no statistical difference" is defined as the null hypothesis, meaning a p-value less than 0.05 would indicate a statistically significant inconsistency between the two. The p-value obtained from the Kruskal-Wallis test for EA-SVR and ENCFIS is 0.0002, while the p-value for Type-1 and ENCFIS is 0.0439, both rejecting the null hypothesis, i.e., the performance of ENCFIS on this dataset is statistically superior to EA-SVR and Type-1. This result suggests that the proposed robust learning method is highly effective, and the introduction of complex fuzzy logic is also immensely positive regarding the accuracy and robustness. Thus, it is sufficient to conclude that the ENCFIS model is robust under high label noise conditions.

TABLE III
COMPARISON OF THE MODELS OVER THE CORRUPTED SYNTHETIC DATA TEST.

	BP	RBF	GRNN	EA-SVR	Type-1	ENCFIS
RMSE	0.639	0.419	0.380	0.126	0.0271	0.0082
Hidden Neurons	5	50	203	-	36	36
Support Vectors	-	-	-	149	-	-
Parameters	16	151	603	-	36	45
Epoch	100	100	100	30	100	60

B. Corrupted Sunspot Time Series Test

Sunspot data records sunspot activity from 1874 to 2022, which can be downloaded from the website of Sunspot Index and Long-term Solar Observations (SILSO) [44]. In this experiment, 602 sunspot observations from 1976 to 1980 are used to create the contaminated training set, while another 602 records from 1984 to 1989 are taken to form the clean validation set. The form of each data point used for the test is designated as $\{\chi(\tau-2), \chi(\tau-1); \chi(\tau)\}$, in which two previous observations are utilized to create the input vector while the current observation $\chi(\tau)$ is considered the output label. For the training set, around 150 observations out of the 602 are evenly replaced with random values, which means 25% of the time series is compromised. This also suggests that if the training data is constructed using the above form, 50% of the data points will contain one incorrect input variable, 25% data points will have a misleading training label and only 25% of the data points are intact. This experiment is far more challenging than the previous one because in the function approximation case only the label noise exists, while the inputs are perfectly preset. But time series forecasting requires to apply the earlier observations to predict the future ones, which gives rise to the inescapable perturbation inside input vectors, bringing an extraordinarily daunting challenge to any algorithm. In other word, all benchmark models suffer from both intense label noises and faulty inputs in this case.

The hyperparameter setting of ENCFIS is shown in TABLE IV, the ccomparison of all benchmark models are given in TABLE V and the visualization of model outputs on both contaminated training set and pure validation set are displayed in Fig.5. As expected, three non-robust algorithms, BP, RBF

TABLE IV
ENCFIS HYPERPARAMETER SETTING FOR THE CORRUPTED SUNSPOT DATA TEST.

Bisecting k-means clustering for antecedence pre-training			
Cluster number k	6	Expansion factor ρ	1
DEMON		M-estimator and Huber Loss	
Learning rate α	10^{-5}	Tuning constant γ	2.9850
Initial momentum factor β	0.3	α-cut factor δ	0.1

and GRNN still perform poorly in this test. Although SVR outperforms non-robust models, it also has obvious deviations in the prediction of peak points. It may be caused by the way SVR sets the "margin" that excludes some correct data points near the peak. Note that the SVR model trained in this test still has 975 support vectors while the training set has 1000 samples, indicating that the SVR algorithm has not obtained the expected sparse representation on this challenging dataset. A model where the number of support vectors is almost the same as the size of the training set is of no practical value. From this perspective, the EA-SVR model does not pass the corrupted Sunspot time series test. Moreover, even the "Type-1" model that uses the same robust learning strategy as ENCFIS, failed to accurately predict the peak part of the time series, which makes ENCFIS the only winner in this challenge. In fact, the Sunspot time series is not a difficult object, and many models can easily achieve good results on this dataset. The only reason for the poor performance is strong noise and massive outliers. Thus, it is fair to say that model robustness against disturbances is the key to excelling in this case. From this perspective, ENCFIS model exhibits outstanding robustness, as it barely gets affected by the outliers, which is not only reflected in Fig. 5, but also verified from the RMSE values on TABLE V. The above factors strongly indicate that the proposed robust learning method contributed to such a result. Nevertheless, given that ENCFIS' type-1 fuzzy counterpart marked as "Type-1" also interfered by the noise, it leads us to the conclusion that the two-dimensional properties of complex fuzzy logic and the disturbance insensitivity it brings is vital to the success of ENCFIS as well.

TABLE V
COMPARISON OF THE MODELS OVER THE CORRUPTED SUNSPOT DATA TEST.

	BP	RBF	GRNN	EA-SVR	Type-1	ENCFIS
RMSE	101.4287	37.5726	75.1519	17.2526	14.1203	6.4720
Hidden Neurons	20	100	1002	-	30	30
Support Vectors	-	-	-	975	-	-
Parameters	81	401	4000	-	30	36
Epoch	100	50	100	30	50	30

C. Ultimate Tensile Strength (UTS) of Metal Alloys Regression Model

The ultimate tensile strength refers to the peak of engineering stress in a stress-strain curve, which is a vital property for metal materials. In metallurgy, it is not easy to ensure materials meet specific requirements, because the relationship between the variables of the production process is highly nonlinear and sparse. In this experiment, 1,500 data samples from actual industrial processes are selected, of which 1,000 were used as training sets and the other 500 were test sets. This dataset has a total of 15 input dimensions, of which 13 are numerical variables and 2 are categorical variables. The specific information of each variable is shown in TABLE VI. It is worth noting that due to the high dimensionality of this dataset, only the 10 most important influencing variables are selected as input information, which are C, Mn, Cr, Mo, Ni, Site, Tempering Temperature, Cooling Medium, Sample Size and Test Depth. In addition, since the data comes directly from industrial processes, it contains an unknown number of outliers, which caused by measurement errors or incorrect human entry, which, albeit in limited numbers, may still offset model performance. It is believed that this experiment can simulate most real-world scenarios, thus, validate the model potential in applications. The hyperparameter setting of the network is shown in TABLE VII, performance indicators of all benchmark models are provided in TABLE VIII and the visualization of the model output is displayed in Fig. 6.

TABLE VI
THE BASICS OF UTS DATA SET.

Numerical Variable	**Mean**	**Median**	**Range**
C	0.3902	0.41	0.12-0.62
Si	0.2546	0.25	0.11-0.35
Mn	0.7524	0.73	0.35-1.72
S	0.021	0.023	0.0005-0.21
Cr	1.053	1.07	0.05-3.46
Mo	0.2631	0.23	0.01-1
Ni	0.8039	0.25	0.02-4.16
Al	0.036	0.027	0.005-1.08
V	0.0075	0.005	0.001-0.27
Hardening Temp	856.81	850	820-980
Tempering Temp	604.18	610	170-730
Sample Size	156.93	150	8-381
Test Depth	16.08	12.7	4-140
UTS	932.09	912.9	516.2-1842
Category symbol			**Categories**
Site			6
Cooling Medium			3

TABLE VII
ENCFIS HYPERPARAMETER SETTING FOR THE UTS DATA TEST.

Bisecting k-means clustering for antecedence pre-training			
Cluster number k	6	Expansion factor ρ	1
DEMON		M-estimator and Huber Loss	
Learning rate α	10^{-6}	Tuning constant γ	3.9104
Initial momentum factor β	0.3	α-cut factor δ	0.2

In terms of RMSE values, the performance of different models on this dataset varies widely. As a result of the high-dimensionality, data sparsity as well as perturbation caused by outliers, traditional non-robust neural network models such as BP, RBF, GRNN, LSTM [45], and DBN [46] have underwhelming performance on this dataset. The classical ANFIS neuro-fuzzy model does not achieve the best prediction performance as well due to the limited expressive ability of the type-1 fuzzy logic. Only three models achieved MSE values that is less than 40, namely EA-SVR, IT2-Sugeno [47] and ENCFIS, and those are close in performance. Among

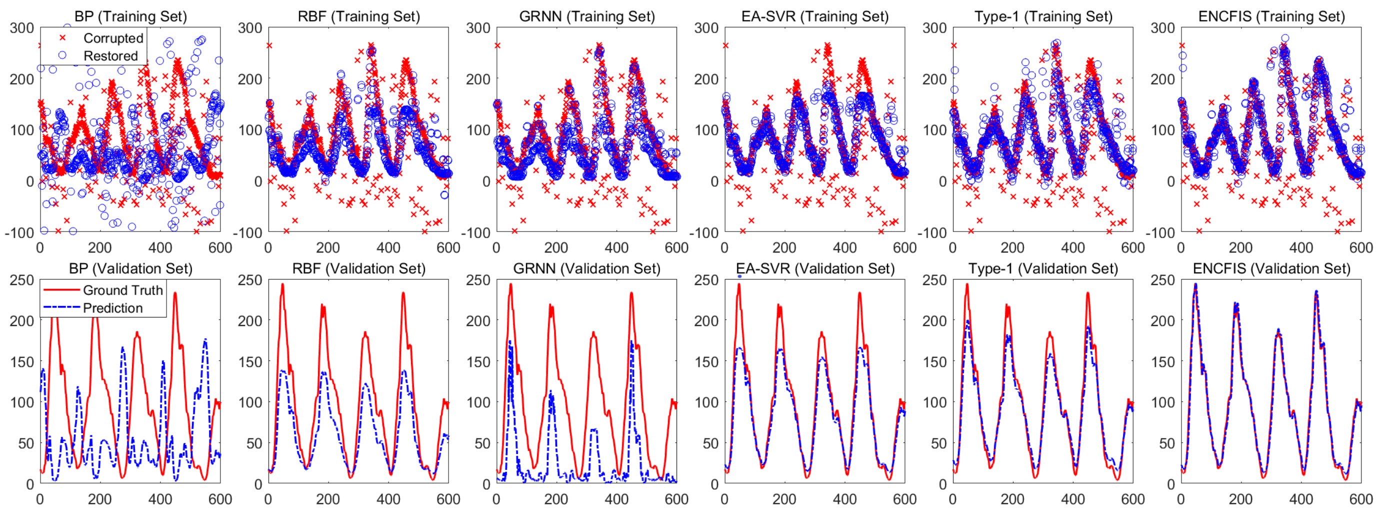

Fig. 5. Model outputs of corrupted Sunspot time series test. (The first row shows the reconstruction results of models on the corrupted training set, and the second row reveals the prediction results of models on the clean validation set.)

them, the good performance of SVR is attributed to its unique vector representation of the input data, which makes it immune to the high dimensionality and sparsity. Also, SVR is robust to noise owing to its setting of "soft margin", which further strengthens its advantages. IT2-Sugeno is a type-2 fuzzy model that benefits from the richer semantic representation of the interval type-2 fuzzy logic. The enhanced generalization capability of the interval type-2 rule-base also increases the reliability, as its parameters are less prone to fluctuations due to occasional outliers. However, both methods have conspicuous disadvantages. The traditional type-2 fuzzy model is not robust enough for heavy noises. The SVR model is overwhelmingly inefficient for nonlinear problems with numerous training samples as the sparse representation is often hard to obtain, and the algorithm is prone to generate excessive support vectors instead. Note that both methods rely on evolutionary algorithms for optimization, which brings the computational complexity a magnitude higher than gradient optimization.

To be more specific on this issue, TABLE VIII also provides the training complexity of each algorithm versus the data dimension m and the number of samples n, where some of these algorithms are sensitive to the former, while others are sensitive to the latter. It can be seen that the EA-SVR model has the training complexity of $O(n^3)$, making it the most n sensitive one among all benchmark models, which explains its unfriendliness against training sets with a large number of data points. For the IT2Sugeno algorithm trained using the evolutionary algorithm, the complexity is $O(mn^2 + n!)$, i.e., it is sensitive to both data dimensionality and sample size, leading to limited application scenarios. In comparison, the complexity of the ENCFIS model, which is $O(mn^2)$, is less sensitive to sample size, giving it a significant advantage over the previous two models in tasks with controlled dimensionality but massive training samples.

Furthermore, ENCFIS obtained the best RMSE performance thanks to the proposed robust learning strategy and the improved generalization capability of complex fuzzy logic. The iteration efficiency of this algorithm is also leading other benchmark algorithms, reflected by the training epochs, as ENCFIS achieves the lowest RMSE value with only 20 iterations. It is fair to say that the proposed ENCFIS is a well-balanced model that integrates performance, efficiency, and robustness, making it highly adaptable to challenging numerical regression tasks and real-world scenarios. It should be mentioned that, unlike most robust models, ENCFIS does not sacrifice its accuracy significantly for the exchange of extra

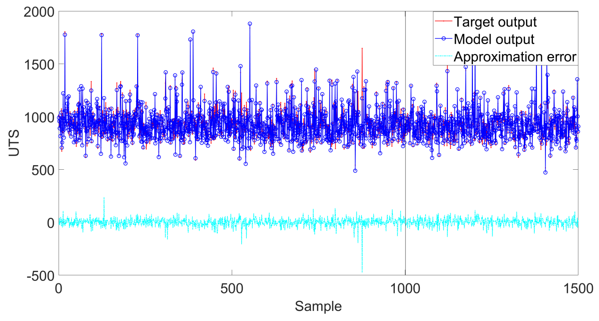

Fig. 6. Performance of ENCFIS on UTS regression. The data samples on the left side of the grey line denote the training set, while the right-side data samples are the test set. Approximation error is displayed in the picture as well.

TABLE VIII
COMPARISON OF THE PERFORMANCE (UTS DATA TEST).

	STD	MAE	SMAPE	RMSE	**Epoch**	**TrainComplex**
BP	150.9674	32.1335	3.4460	44.4965	200	$O(mn)$
RBF	149.2079	41.2173	4.4341	54.1319	100	$O(mn^2)$
GRNN	153.8146	39.2092	4.1841	56.5168	100	$O(mn^2)$
LSTM	137.5868	42.4865	4.5160	56.5765	200	$O(mn^2 + m^2n)$
DBN	142.1892	36.1535	3.9260	47.7999	50	$O(mk^{m+1})$
EA-SVR	153.1510	29.5739	3.1759	39.5723	30	$O(n^3)$
IT2Sugeno	-	-	-	38.7600	100	$O(mn^2 + n!)$
ANFIS	148.6884	36.4036	3.9672	45.4163	50	$O(mn^2)$
ENCFIS	153.6219	28.8886	3.1456	38.0137	20	$O(mn^2)$

robustness, which suggests that excellent robustness and good model performance can co-exist in the same architecture.

V. Conclusions and Future Work

In this paper, the first robust machine learning architecture based on complex fuzzy theory for numerical regression purpose is proposed. This algorithm has a unique robust learning strategy to deal with noise and outliers, i.e., firstly it utilizes clustering of input variables to pretrain kernel parameters to make them close to the optimal position, then further adopts momentum decay gradient method, Huber loss and Welsch M-estimator as a joint robust learning method to accurately predict parameters under the condition of massive noise interference. At the same time, the two-dimensional attributes as well as richer information capacity of the complex fuzzy rule-base also increase the generalization capability and anti-disturbance performance of the algorithm. In addition, the existence of closed-form solutions for the first derivatives of complex fuzzy membership functions enables gradient optimization, thereby increasing algorithm efficiency, which is a significantly superiority over type-2 fuzzy architectures that gradient optimization is unavailable. The experimental results also confirm that the proposed ENCFIS algorithm exhibits amazing adaptability whether for a problem with only label noise or when both input variables and labels are noisy. Therefore, it has reason to believe that such design is very successful as a robust learning attempt.

Future work should mainly include the following aspects. The first is that we will continue to improve the current architecture, especially to find a better robust objective function to replace Huber loss, to avoid the trouble of determining the additional hyperparameter. Secondly, we will search for a more effective optimization policy in order to obtain better optimization results in intricate real-world scenarios. Thirdly, we will further study the complex fuzzy theory and develop better complex membership functions to replace the current solution which is rather primitive. In addition, we are also considering introducing an outlier detection mechanism to identify outliers within the input variables. Given that deeper networks are currently the mainstream in this direction, we will also conduct research in this regard to improve robustness as well as performance by increasing the depth of the network.

References

[1] D. Freedman, "Statistical models: theory and practice, second edition," 2009.

[2] J. Jang and J. Roger, "Anfis adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems Man and Cybernetics*, vol. 23, no. 3, pp. 665–685, 1993.

[3] L.-X. Wang, "The wm method completed: a flexible fuzzy system approach to data mining," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 6, pp. 768–782, 2003.

[4] C. Chao, R. John, J. Twycross, and J. M. Garibaldi, "Type-1 and interval type-2 anfis: A comparison," in *IEEE International Conference on Fuzzy Systems*, 2017, pp. 1–6.

[5] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009.

[6] S. M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 86–94, 2017.

[7] N. Papernot, P. Mcdaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM Asia Conference on Computer and Communications Security*, 2017.

[8] M. Shafique, M. Naseer, T. Theocharides, C. Kyrkou, O. Mutlu, L. Orosa, and J. Choi, "Robust machine learning systems: Challenges, current trends, perspectives, and the road ahead," *IEEE design and test*, vol. 37, no. 2, pp. 30–57, 2020.

[9] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," in *Advances in Neural Information Processing Systems*, M. Mozer, M. Jordan, and T. Petsche, Eds., vol. 9. MIT Press, 1996. [Online]. Available: https://proceedings.neurips.cc/paper/1996/file/d38901788c533e8286cb6400b40b386d-Paper.pdf

[10] B. Smola, A. J.and Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[11] P. J. Huber, "Robust estimation of a location parameter," *The Annals of mathematical statistics*, vol. 35, no. 1, pp. 73–101, 1964.

[12] Q. Wang, Y. Ma, K. Zhao, and Y. Tian, "A comprehensive survey of loss functions in machine learning," *Annals of Data Science*, vol. 9, no. 2, pp. 187–212, 2020.

[13] J. T. Barron, "A general and adaptive robust loss function," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4326–4334.

[14] B. Han, Q. Yao, T. Liu, G. Niu, I. W. Tsang, J. T. Kwok, and M. Sugiyama, "A survey of label-noise representation learning: Past, present and future." *arXiv*, 2020.

[15] J. Goldberger and E. Benreuven, "Training deep neural-networks using a noise adaptation layer," *ICLR*, 2016.

[16] G. Patrini, A. Rozza, A. Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: a loss correction approach," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2233–2241, 2016.

[17] Y. Wang, A. Kucukelbir, and D. M. Blei, "Robust probabilistic modeling with bayesian data reweighting," *Proceedings of the 34th International Conference on Machine Learning*, vol. 70, pp. 3646–3655, 2017.

[18] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," *ICML*, 2018.

[19] J. Lu, Z. Zhou, T. Leung, L. J. Li, and F. L. Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *ICML 2018*, 2018.

[20] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," *NeurIPS*, pp. 8535–8545, 2018.

[21] Z. Wang, G. Hu, and Q. Hu, "Training noise-robust deep neural networks via meta-learning," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 4523–4532.

[22] Y. Carmon, A. Raghunathan, L. Schmidt, J. C. Duchi, and P. S. Liang, "Unlabeled data improves adversarial robustness," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/32e0bd1497aa43e02a42f47d9d6515ad-Paper.pdf

[23] S. Dick and O. Yazdanbakhsh, "A systematic review of complex fuzzy sets and logic," *Fuzzy Sets & Systems*, vol. 338, no. MAY1, pp. 1–22, 2018.

[24] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," *KDD workshop on text mining*, vol. 400, pp. 525–526, 2000.

[25] D. Menezes, D. M. Prata, A. R. Secchi, and J. C. Pinto, "A review on robust m-estimators for regression analysis," *Computers & Chemical Engineering*, vol. 147, no. 8, p. 107254, 2021.

[26] D. Ramot, R. Milo, M. Friedman, and A. Kandel, "Complex fuzzy sets," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 2, pp. 171–186, 2002.

[27] Ramot, D., Friedman, M., Langholz, G., Kandel, and A., "Complex fuzzy logic," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 4, pp. 450–461, 2003.

[28] S. Dick, "Toward complex fuzzy logic," *IEEE Transactions on Fuzzy Systems*, vol. 13, no. 3, pp. 405–414, 2005.

[29] J. Y. Man, Z. Chen, and S. Dick, "Towards inductive learning of complex fuzzy inference systems," in *Fuzzy Information Processing Society, Nafips 07 Meeting of the North American*, 2007.

[30] C. Li, T. W. Chiang, J. W. Hu, and T. Wu, "Complex neuro-fuzzy intelligent approach to function approximation," in *Third International Workshop on Advanced Computational Intelligence*, 2010.

[31] R. Shoorangiz and M. H. Marhaban, “Complex neuro-fuzzy system for function approximation,” in *International Journal of Applied Electronics in Physics & Robotics*, vol. 1, no. 2, 2013, pp. 5–9.
[32] P. J. Huber, *Robust Statistics*, M. Lovric, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. [Online]. Available: https://doi.org/10.1007/978-3-642-04898-2_594
[33] H. Theil, *A Rank-Invariant Method of Linear and Polynomial Regression Analysis*. Dordrecht: Springer Netherlands, 1992, pp. 345–381. [Online]. Available: https://doi.org/10.1007/978-94-011-2546-8_20
[34] P. K. Sen, “Estimates of the regression coefficient based on kendall’s tau,” *Journal of the American Statistical Association*, vol. 63, no. 324, pp. 1379–1389, 1968. [Online]. Available: http://www.jstor.org/stable/2285891
[35] M. Sugeno and G. Kang, “Structure identification of fuzzy model,” *Fuzzy Sets and Systems*, vol. 28, no. 1, pp. 15–33, 1988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0165011488901133
[36] J. Chen, C. Wolfe, Z. Li, and A. Kyrillidis, “Demon: Improved neural network training with momentum decay,” in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 3958–3962.
[37] S. Ruder, “An overview of gradient descent optimization algorithms,” *CoRR*, vol. abs/1609.04747, 2016. [Online]. Available: http://arxiv.org/abs/1609.04747
[38] E. Kreyszig, H. Kreyszig, and E. J. Norminton, *24.1 Data Representation. Average. Spread*. Hoboken, N.J.: Wiley, 2011, pp. 1011–1015.
[39] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” vol. 323, no. 6088, pp. 533–586, 1988.
[40] J. Moody and C. Darken, “Fast learning in networks of locally-tuned processing units,” *Neural Computation*, vol. 1, no. 2, pp. 281–294, 1989.
[41] D. F. Specht, “A general regression neural network,” *IEEE Transactions Neural Networks*, vol. 2, no. 6, pp. 568–76, 1991.
[42] M. Jaderberg, V. Dalibard, S. Osindero, W. M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan, C. Fernando, and K. Kavukcuoglu, “Population based training of neural networks,” *CoRR*, vol. abs/1711.09846, 2017. [Online]. Available: http://arxiv.org/abs/1711.09846
[43] W. H. Kruskal and W. A. Wallis, “Use of ranks in one-criterion variance analysis,” *Journal of the American Statistical Association*, vol. 47, no. 260, pp. 583–621, 1952. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/01621459.1952.10483441
[44] *Sunspot Index and Long-term Solar Observations (SILSO)*. [Online]. Available: https://www.sidc.be/silso/datafiles
[45] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, pp. 1735–1780, 12 1997.
[46] G. E. Hinton, “Deep belief networks,” *Scholarpedia*, vol. 4, no. 5, p. 5947, 2009.
[47] Q. Liang and J. Mendel, “An introduction to type-2 tsk fuzzy logic systems,” in *FUZZ-IEEE’99. 1999 IEEE International Fuzzy Systems. Conference Proceedings (Cat. No.99CH36315)*, vol. 3, 1999, pp. 1534–1539 vol.3.