

Article

# Correntropy-Based Constructive One Hidden Layer Neural Network

Mojtaba Nayyeri <sup>1</sup>, Modjtaba Rouhani <sup>2</sup> , Hadi Sadoghi Yazdi <sup>2</sup>, Marko M. Mäkelä <sup>3</sup>, Alaleh Maskooki <sup>3</sup> and Yury Nikulin <sup>3,\*</sup> 

<sup>1</sup> Institute for Artificial Intelligence, University of Stuttgart, 70569 Stuttgart, Germany; mojtaba.nayyeri@ki.uni-stuttgart.de

<sup>2</sup> Computer Engineering Department, Ferdowsi University of Mashhad, Mashhad 1696700, Iran

<sup>3</sup> Department of Mathematics and Statistics, University of Turku, 20014 Turku, Finland

\* Correspondence: yurnik@utu.fi

**Abstract:** One of the main disadvantages of the traditional mean square error (MSE)-based constructive networks is their poor performance in the presence of non-Gaussian noises. In this paper, we propose a new incremental constructive network based on the correntropy objective function (correntropy-based constructive neural network (C2N2)), which is robust to non-Gaussian noises. In the proposed learning method, input and output side optimizations are separated. It is proved theoretically that the new hidden node, which is obtained from the input side optimization problem, is not orthogonal to the residual error function. Regarding this fact, it is proved that the correntropy of the residual error converges to its optimum value. During the training process, the weighted linear least square problem is iteratively applied to update the parameters of the newly added node. Experiments on both synthetic and benchmark datasets demonstrate the robustness of the proposed method in comparison with the MSE-based constructive network, the radial basis function (RBF) network. Moreover, the proposed method outperforms other robust learning methods including the cascade correntropy network (CCOEN), Multi-Layer Perceptron based on the Minimum Error Entropy objective function (MLPMEE), Multi-Layer Perceptron based on the correntropy objective function (MLPMCC) and the Robust Least Square Support Vector Machine (RLS-SVM).

**Keywords:** information theoretic learning; probability theory; measure space; correntropy; non-Gaussian noise; constructive network; compact architecture; half-quadratic programming problem



**Citation:** Nayyeri, M.; Rouhani, M.; Yazdi, H.S.; Mäkelä, M.M.; Maskooki, A.; Nikulin, Y. Correntropy-Based Constructive One Hidden Layer Neural Network.

*Algorithms* **2024**, *17*, 49. <https://doi.org/10.3390/a17010049>

Academic Editor: Takeshi Yamada

Received: 29 November 2023

Revised: 9 January 2024

Accepted: 11 January 2024

Published: 22 January 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Non-Gaussian noises, especially impulse noise, and outliers are one of the most challenging issues in training adaptive systems including adaptive filters and feedforward networks (FFNs). The mean square error (MSE), the second-order statistic, is used widely as the objective function for adaptive systems due to its simplicity, analytical tractability and linearity of its derivative. The Gaussian noise assumption beyond MSE objective functions supposes that many real-world random phenomena may be modeled by Gaussian distribution. Under this assumption, MSE could be capable of extracting all information from data whose statistic is defined solely by the mean and variance [1]. Most real-world random phenomena do not have a normal distribution and the MSE-based methods may perform unsatisfactorily in such cases.

Several types of feedforward networks have been proposed by researchers. From the architecture viewpoint, these networks can be divided into four classes including fixed structure networks, constructive networks [2–6], pruned networks [7–10] and pruning constructive networks [11–13].

The constructive networks start with a minimum number of nodes and connections, and the network size is increased gradually. These networks may have an adjustment mechanism based on the optimization of an objective function. The following literature survey focuses on the single-hidden layer feedforward networks (SLFNs) and multi-hidden layer feedforward networks with incremental constructive architecture, which are trained based on the MSE objective function.

Fahlman and LeBrier [2] proposed a cascade correlation network (CCN) in which new nodes are added and trained one by one, creating a multi-layer structure. The parameters of the network are trained to maximize the correlation between the output of the new node and the residual error. The authors in [3] proposed several objective functions for training the new node. They proved that the networks with such objective functions are universal approximators. Huang et al. [4] proposed a novel cascade network. They used the orthogonal least square (OLS) method to drive a novel objective function for training new hidden nodes. Ma and Khorasani [6] proposed a constructive one hidden layer feedforward network in which its hidden unit activation functions are Hermite polynomial functions. This approach results in a more efficient capture of the underlying input–output map. They proposed a new one hidden layer constructive adaptive neural network (OHLCN) scheme in which the input and output sides of the training are separated [5]. They scaled error signals during the learning process to achieve better performance. Inefficient input connections are pruned to achieve better performance. A new constructive scheme was proposed by Wu et al. [14] based on a hybrid algorithm, which is presented by combining the Levenberg–Marquardt algorithm and the least square method. In their approach, a new randomly selected neuron is added to the network when training is entrapped into local minima.

Inspired by information theoretic learning (ITL), correntropy, which is a localized similarity measure between two random variables [15,16], has recently been utilized as the objective function for training adaptive systems. Bessa et al. [17] employed maximum correntropy criterion (MCC) for training neural networks with fixed architecture. They compared the Minimum Error Entropy (MEE) and MCC-based neural networks with MSE-based networks and reported new results in wind power prediction. Singh and Principe [18] used correntropy as the objective function in the linear adaptive filter to minimize the error between the output of the adaptive filter and the desired signal, to adjust the filter weights. Shi and Lin [19] employed a convex combination scheme to improve the performance of the MCC adaptive filtering algorithm. They showed that the proposed method has better performance compared to the original signal filter algorithm. Zhao et al. [20] combined the advantage of Kernel Adaptive Filter and MCC and proposed Kernel Maximum Correntropy (KMC). The simulation results showed that KMC has significant performance in the noisy frequency doubling problem [20]. Wu et al. [21] employed MCC to train Hammerstein adaptive filters and showed that it provides a robust method in comparison to the traditional Hammerstein adaptive filters. Chen et al. [22] studied a fixed-point algorithm for MCC and showed that under sufficient conditions convergence of the fixed-point MCC algorithm is guaranteed. The authors in [23] studied the steady-state performance of adaptive filtering when MCC is employed. They established a fixed-point equation in the Gaussian noise condition to obtain the exact value of the steady-state excess mean square error (EMSE). In non-Gaussian conditions, using the Taylor expansion approach, they derived an approximate analytical expression for the steady-state EMSE. Employing stack auto-encoders and the correntropy-induced loss function, Chen et al. [24] proposed a robust deep learning model. The authors in [25], inspired by correntropy, proposed a margin-based loss function for classification problems. They showed that in their method, outliers that produce high error have little effect on discriminant function. In [26], the authors provided a learning theory analysis for the connection between the regression model associated with the correntropy-induced loss and the least square regression model. Furthermore, they studied its convergence property and concluded that the scale parameter provides a balance between the convergence rate of the model and its robustness. Chen and

Principe [27] showed that maximum correntropy estimation is a smooth maximum a posteriori estimation. They also proved that when kernel size is larger than the special value and some condition is held, maximum correntropy estimation has a unique optimal solution due to the strictly concave region of the smooth posterior distribution. The authors in [28], investigated the approximation ability of a cascade network when its input parameters are calculated by the correntropy objective function with a sigmoid kernel. They reported that their method works better than the other methods introduced in [28] when data are contaminated by noise.

MCC with Gaussian kernel is a non-convex objective function that leads to local solutions for neural networks. In this paper, we propose a new method to overcome this bottleneck by adding hidden nodes one by one until the constructive network reaches a specific amount of predefined accuracy or reaches a maximum number of nodes. We prove that the correntropy of the constructive network constitutes a strictly increasing sequence after adding each hidden node and converging to its maximum.

This paper can be considered as an extension of [28]. While in [28] the correntropy measure was based on the sigmoid kernel in the objective function to adjust the input parameters of a newly added node in a cascade network, in this paper, the kernel in the correntropy objective function is changed from sigmoid to Gaussian kernel. This objective function is then used for training both input and output parameters of the new nodes in a single-hidden layer network. The proposed method performs better than [28] for two reasons: (1) the Gaussian kernel provides better results than the sigmoid kernel as it is a local similarity measure, and (2) in contrast to [28], in this paper, correntropy is used to train both the input and output parameters of each newly added node.

In a nutshell, the proposed method has the following advantages:

1. The proposed method is robust to non-Gaussian noises, especially impulse noise, since it takes advantage of the correntropy objective function. In particular, the Gaussian kernel provides better results than the sigmoid kernel. The reason for the robustness of the proposed method is discussed in Section 4 analytically, and in Section 5 experimentally.
2. Most of the methods that employ correntropy as the objective function to adjust their parameters suffer from local solutions. In the proposed method, the amount of correntropy of the network is increased by adding new nodes and converging to its maximum; thus, the global solution is provided.
3. The network size is determined automatically; consequently, the network does not suffer from over/underfitting, which results in satisfactory performance.

The structure of the remainder of this paper is as follows. In Section 2, some necessary mathematical notations, definitions and theorems are presented. Section 3 presents some related previous work. Then a correntropy-based constructive neural network (C2N2) is proposed in Section 4. Experimental results and a comparison with other methods are carried out in Section 5. The paper is concluded in Section 6.

## 2. Mathematical Notations, Definitions and Preliminaries

In this section, first, measure and function spaces that are necessary for describing previous work are defined in Section 2.1. Section 2.2 introduces the structure of the single-hidden layer feedforward network (SLFN) that is used in this paper, followed by its mathematical notations and definitions of its related variables.

2.1. Measure Space, Probability Space and Function Space

As mentioned in [3], let  $\mathcal{X}$  be the input space that is a bounded measurable subset in  $\mathbb{R}^d$  and  $L^2(\mathcal{X})$  be the space of all function  $f$  that is  $\int_{\mathcal{X}} (f(x))^2 d\mu(x) < \infty$ . For  $u, v \in L^2(\mathcal{X})$ , the inner product is defined as follows:

$$\langle u, v \rangle_{\mu} := \int_{\mathcal{X}} u(x)v(x)d\mu(x).$$

where  $\mu$  is a positive measure on input space. Under the measure  $\mu$ , the  $l_2$  norm in  $L^2(\mathcal{X})$  space is denoted as  $|\cdot|_2$ . The closeness between  $u$  and  $v$  is measured by

$$|u - v|_2 = \left( \int_{\mathcal{X}} (u(x) - v(x))^2 d\mu(x) \right)^{\frac{1}{2}}$$

The angle between  $u$  and  $v$  is defined by

$$\theta_{u,v} := \arccos \left( \frac{\langle u, v \rangle_{\mu}}{|u|_2 |v|_2} \right)$$

**Definition 1 ([28,29]).** Let  $W$  be a probability space that is a measure space with a total measure one. This space is represented as follows:

$$W = (\Omega, \mathcal{F}, \mathcal{P})$$

where  $\Omega$  is its sample space. In this paper,  $\Omega$  is considered a compact subset of  $\mathbb{R}^d$ ,  $\mathcal{F}$  is a sigma-algebra of events and  $\mathcal{P}$  is a probability measure that is a measure on  $\mathcal{F}$  with  $\mathcal{P}(\Omega) = 1$ .

**Definition 2 ([28,29]).** Let  $\mathcal{L}^p(\Omega, \mathcal{F}, \mathcal{P})$ ,  $1 \leq p < \infty$  be a set of all  $p$ -integrable random variables  $X : \Omega \rightarrow \mathbb{R}$ , i.e.,

$$\|X\|_p = \left( \int_{\Omega} X^p d\mathcal{P} \right)^{\frac{1}{p}} = (E(X^p))^{\frac{1}{p}} < \infty$$

This is a vector space and the inner product in this space is defined as follows:

$$\langle X, Y \rangle := \int_{\Omega} X(\omega)Y(\omega)d\mathcal{P} = E(XY)$$

where  $X, Y \in \mathcal{L}^p(\Omega, \mathcal{F}, \mathcal{P})$  and  $E(\cdot)$  is expectation in probability theory. The closeness between two random variables  $X$  and  $Y$  is measured by  $\mathcal{L}^p(\Omega, \mathcal{F}, \mathcal{P})$  norm:

$$\|X - Y\|_p = \left( \int_{\Omega} |X(\omega) - Y(\omega)|^p d\mathcal{P} \right)^{\frac{1}{p}} = (E(|X - Y|^p))^{\frac{1}{p}} \quad X, Y \in \mathcal{L}^p(\Omega, \mathcal{F}, \mathcal{P})$$

In ITL, the correlation between random variables is generalized to correntropy, which is a measure of similarity [15]. Let  $X$  and  $Y$  be two given random variables; the correntropy in the sense of [15] is defined as

$$V_{\mathcal{M}}(X, Y) := E(k_{\mathcal{M}}(X, Y))$$

where  $k_{\mathcal{M}}(\cdot, \cdot)$  is a Mercer kernel function.

In general, a Mercer kernel function is a type of positive semi-definite kernel function that satisfies Mercer’s condition. Formally, a symmetric function  $k_M$  is a Mercer kernel function if, for any positive integer  $m$  and any set of random variables  $X_1, \dots, X_m \in \mathcal{L}^p(\Omega, \mathcal{F}, \mathcal{P})$ , the corresponding Gram matrix  $K_{ij} = k_M(X_i, X_j)$  is positive semi-definite.

In the definition of correntropy,  $E(\cdot)$  implies the expected value of the random variable and  $\mathcal{M}$  is replaced by  $\alpha$  or  $\sigma$  if the sigmoid or Gaussian kernel (radial basis) are used, respectively. In our recent work [28], we use a sigmoid kernel, which is defined as

$$k_\alpha(X, Y) = \tanh(\alpha \langle X, Y \rangle + c),$$

where  $\alpha, c \in \mathcal{R}$  are scale and offset hyperparameters of the sigmoid kernel. The offset parameter  $c$  in the sigmoid kernel influences the shape of the kernel function. A higher value of  $c$  leads to a steeper sigmoid curve, making the kernel function more sensitive to variations in the input space. It is important to note that the choice of hyperparameters, including  $c$  and  $\alpha$ , can significantly impact the performance of a machine learning model using the sigmoid kernel. These parameters are often tuned during the training process to optimize the model for a specific task or dataset.

In contrast to [28], in this paper, we use the Gaussian kernel that is represented as

$$k_\sigma(X, Y) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|X - Y\|^2}{2\sigma^2}\right)$$

where  $\sigma$  is the variance for the Gaussian function. Here,  $\sigma$  controls the width of the Gaussian kernel. A larger  $\sigma$  results in a smoother and more slowly decaying kernel, while a smaller  $\sigma$  leads to a narrower and more rapidly decaying kernel.

Let error function be defined as  $e := e(X, Y) \equiv X - Y$ ; the correntropy of the error function is represented as

$$V_\sigma(e) = E(k_\sigma(X, Y)).$$

At the end of this subsection, we note that, alternatively, the Wasserstein distance, also known as the Earth Mover's Distance (EMD), Kantorovich–Rubinstein metric, Mallows's distance or optimal transport distance, can be used as a metric that quantifies the minimum cost of transforming one probability distribution into another and can therefore be used to quantify the rate of convergence when the error is measured in some Wasserstein distance [30]. The relationship between correntropy and Wasserstein distance is often explored in the context of kernelized Wasserstein distances. By using a kernel function, the Wasserstein distance can be defined in a reproducing kernel Hilbert space (RKHS). In this framework, correntropy can be seen as a special case of a kernelized Wasserstein distance when the chosen kernel is the Gaussian kernel.

## 2.2. Network Structure

This paper focuses on the single-hidden layer feedforward network. As shown in Figure 1, it has three layers, including the input layer, the hidden layer and the output layer. Without loss of generality, this paper considers the SLFN with only one output node.

The output of SLFN with  $L$  hidden nodes is represented as follows [3]:

$$f_L = \sum_{i=1}^L \beta_i g_i(x)$$

where  $g_i$  is the  $i$ -th hidden node and can be one of the two following types:

1. For additive nodes

$$g_i(x) = g(\langle w_i, x \rangle_\mu + b_i); w_i \in \mathbb{R}^d \text{ and } b_i \in \mathbb{R}$$

2. For RBF nodes

$$g_i(x) = g\left(\frac{\|x - w_i\|_2}{b_i}\right); w_i \in \mathbb{R}^d \text{ and } b_i \in \mathbb{R}^+$$

For additive nodes, the vector  $w_i$  is the input weights of the  $i$ -th hidden node and  $b_i$  is its bias. For RBF nodes, the vector  $w_i$  is the center of the  $i$ -th radial basis function and  $b_i$  is its impact factor.

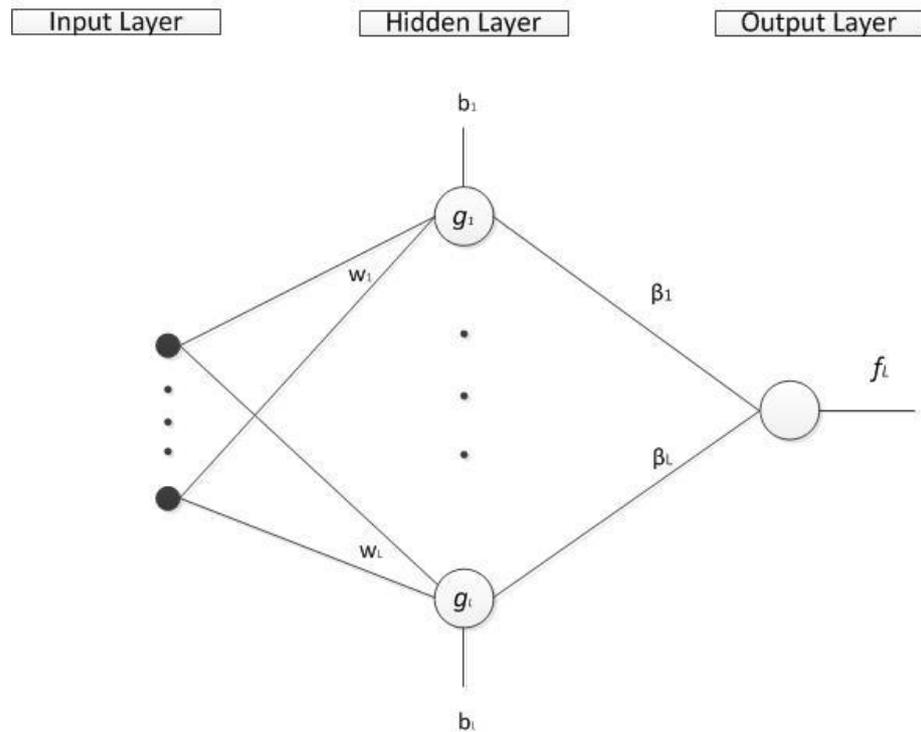


Figure 1. SLFN with additive nodes.

All networks that can be generated are represented as the following functions set [3]:

$$\mathcal{O} = \bigcup_{L=1}^{\infty} \mathcal{O}_L$$

where

$$\mathcal{O}_L = \left\{ f_L \mid f_L(x) = \sum_{i=1}^L \beta_i g_i(x); \beta_i \in \mathbb{R}, g_i(x) \in \mathcal{G} \right\}$$

and  $\mathcal{G}$  is a set of all possible hidden nodes. For additive nodes, we have

$$\mathcal{G} = \left\{ g(\langle w, x \rangle_{\mu} + b); w \in \mathbb{R}^d, b \in \mathbb{R} \right\}$$

For the RBF case, we have

$$\mathcal{G} = \left\{ g\left(\frac{\|x - w\|_2}{b}\right); w \in \mathbb{R}^d, b \in \mathbb{R}^+ \right\}$$

Let  $f$  be a target function that is approximated by the network with  $L$  hidden nodes. The network residual error function is defined as follows:

$$e_L := f - f_L,$$

In practice, the function form of error is not available and the network is trained on finite data samples, which are described as  $\mathcal{X} = \{x_i, y_i\}_{i=1}^N$ , where  $x_i \in \mathbb{R}^d$  is the  $d$  dimension input vector of the  $i$ -th training sample and  $y_i \in \mathbb{R}$  is its target value. Thus, the error vector on the training samples is denoted as follows:

$$E_L = (E_{L1}, \dots, E_{LN}),$$

where  $E_{Li}$  is the error of the  $i$ -th training sample for the network with  $L$  hidden nodes ( $E_{Li} = e_L(x_i)$ ). Furthermore, the activation vector for the  $L$ -th hidden node is

$$G_L = (G_{L1}, \dots, G_{LN})$$

where  $G_{Li}$  is the output of the  $L$ -th hidden nodes for the  $i$ -th training sample ( $G_{Li} = g_L(x_i)$ ).

### 3. Previous Work

There are several types of constructive neural networks. In this section, the networks that are proposed in [3,28] are introduced. In those methods, the network is constructed by adding a new node to the network in each step. The training process of the newly added node ( $L$ -th hidden node) is divided into two phases: the first phase is devoted to adjusting the input parameters and the second phase is devoted to adjusting the output weight. When the parameters of the new node are obtained, they are fixed and do not change during the training of the next nodes.

#### 3.1. The Networks Introduced in [3]

For the input parameters' ( $w_L, b_L$ ) adjustment in [3], several objective functions are proposed to adjust the input parameters of the newly added node in the constructive network. They are as follows [3]:

$$\begin{aligned} V_1 &= \left( \frac{E_{L-1}G_L^T}{G_L G_L^T} \right)^2, \\ V_2 &= \left( E_{L-1}G_L^T \right)^2, \\ V_3 &= \left( \frac{(E_{L-1} - \bar{E}_{L-1})(G_L - \bar{G}_L)^T}{\|G_L - \bar{G}_L\|} \right)^2, \\ V_4 &= \sqrt{V_1}, \\ V_5 &= \sqrt{V_2}, \\ V_6 &= \sqrt{V_3}, \\ V_{\text{CasCor}} &= \left( (E_{L-1} - \bar{E}_{L-1})(G_L - \bar{G}_L)^T \right)^2, \end{aligned}$$

where  $V_{\text{CasCor}}$  is the objective function for the cascade correlation network,  $\bar{E}_{L-1} = \frac{1}{N} \sum_{i=1}^N E_{L-1}(x_i)$ . The objective function that is used to adjust the output weight of the  $L$ -th hidden node is [3]

$$\Delta_L = E_{L-1}E_{L-1}^T - E_L E_L^T$$

and  $\Delta_L$  is maximized if and only if [3]

$$\beta_L = \frac{E_{L-1}G_L^T}{G_L G_L^T}$$

which is the optimum output parameter of the new node. In [3], the authors also proved that for each of the objective functions  $V_1$  to  $V_6$ , the network error converges.

**Theorem 1 ([3]).** *Given span  $(G)$  is dense in  $L^2$  and  $\forall g \in \mathcal{G}, 0 < \|g\|_2 < b$  for some  $b \in \mathbb{R}$ . If  $g_L$  is selected so as to maximize  $\left( \frac{(e_{L-1}, g_L)_H}{\|g_L\|_2} \right)^2$ , then  $\lim_{L \rightarrow \infty} \|f - f_L\|_2 = 0$ .*

More detailed discussion about theorems and their proofs can be found in [3].

### 3.2. Cascade Correntropy Network (CCOEN) [28]

The authors in [28] proved that if the input parameters of each new node in a cascade network are assessed by using the correntropy objective function with the sigmoid kernel and its output parameter is adjusted by

$$\beta_L = \frac{E(e_{L-1}g_L)}{E(g_L^2)}$$

then the network is a universal approximator. The following theorem investigates the approximation ability of CCOEN:

**Theorem 2 ([28]).** *Suppose  $\text{span}(g)$  is dense in  $\mathcal{P}^2(\Omega, \mathcal{F}, \mathcal{G})$ . For any continuous function  $f$  and for the sequence of error similarity feedback functions  $\{g_L^{s(e)}\}, L \in \mathbb{N}$ , there exists a real sequence  $\{\eta_L; L \in \mathbb{N}\}$  such that*

$$\lim_{L \rightarrow \infty} E(e_L^2) = 0$$

holds with probability one if

$$g_L^{s(e)} = \operatorname{argmax}_{g_L \in \mathcal{G}} V_\sigma(e_{L-1}, \eta_L g_L)$$

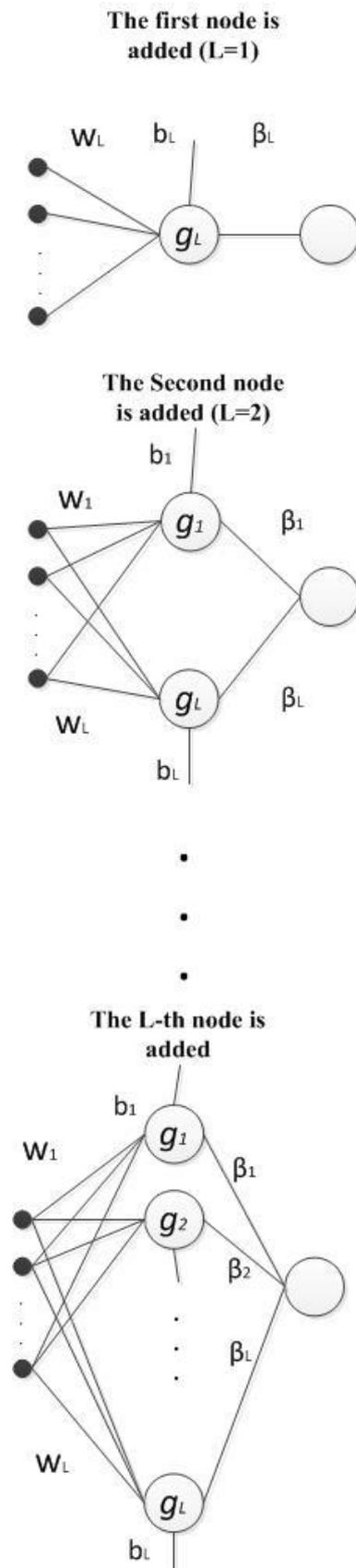
$$\beta_L = \frac{E(e_{L-1}g_L^{s(e)})}{E((g_L^{s(e)})^2)}$$

It was shown that CCOEN is more robust than the networks proposed in [3] when data are contaminated by noise.

## 4. Proposed Method

In this section, a novel constructive neural network is proposed based on the maximum correntropy criterion with the Gaussian kernel. To the best of our knowledge, it is the first time that correntropy with Gaussian kernel is employed as the objective function for training both the input and output weights of a single-hidden layer constructive network. It must be considered that the correntropy is a non-convex objective function and it is difficult to adjust the optimum solution. This section proposes a new theorem and surprisingly proves that the proposed method that is trained by using the correntropy objective function converges to the global solution. It is shown that the performance of the proposed method is excellent in the presence of non-Gaussian noise, especially impulse noise. In the proposed network, hidden nodes are added and trained one by one and the parameters of the newly added ( $L$ -th hidden node) nodes are obtained and then fixed (see Figure 2).

This section is organized as follows: First, some preliminaries, mathematical definitions and theorems that are necessary for presenting the proposed method and proving the convergence of the method are introduced in Section 4.1. The new training strategy for the proposed method is described in Section 4.2. In Section 4.3, the convergence of the proposed method is proven when the error and activation function are continuous random variables. In practice and during the training on the dataset, the error function is not available; thus, the error vector and activation vector are used to train the new node. Regarding this fact, in Section 4.1, two optimization problems are presented to adjust the parameters of the new node based on training data samples.



**Figure 2.** Constructive network in which the last added node is referred to by  $L$ .

#### 4.1. Preliminaries for Presenting the Proposed Method

This section presents a new theorem for the proposed method based on special spaces, which are defined in Definitions 1 and 2.

The following lemmas, propositions and theorems are also used in the proof of the main theorem.

**Lemma 1** ([31]). *Given  $g : R \rightarrow R, \text{Span}\{g(\langle w, x \rangle + b), (w, b) \in \mathbb{R}^d \times \mathbb{R}\}$  is dense in  $L^p$  for every  $p \in [1, \infty)$ , if and only if  $g$  is not a polynomial (almost everywhere).*

**Proposition 1** ([32]). *For  $G(z) = \exp\left(-\frac{\|z\|^2}{2\sigma^2}\right)$ , there exists a convex conjugated function  $\phi$ , such that*

$$G(z) = \sup_{\alpha \in \mathcal{R}^-} \left( \alpha \frac{\|z\|^2}{2\sigma^2} - \phi(\alpha) \right)$$

*Moreover, for a fixed  $z$ , the supremum is reached at  $\alpha = -G(z)$ .*

**Theorem 3** ([33]). *If  $X_n$  is any sequence of random variables which are positive (take values in  $[0, \infty)$ , increasingly converge ( $X_n(\omega) \uparrow X(\omega)$ ) for any  $\omega \in \Omega$ ), and the expectation exists ( $X_n \in L^1$  for all  $n$ ), then  $E(X_n) \rightarrow E(X)$ .*

**Theorem 4** ([33]). *(Monotonicity) Let  $X$  and  $Y$  be random variables with  $X \leq Y$ , then  $E(X) \leq E(Y)$ , with equality if and only if  $X = Y$  almost surely.*

**Theorem 5** ([34]). *(Convergence) Every upper bounded increasing sequence converges to its supremum.*

#### 4.2. C2N2: Objective Function for Training the New Node

In this subsection, we combine the idea of constructive SLFN with the idea of correntropy and propose a new strong constructive network that is robust to impulsive noise. The proposed method employs correntropy as the objective function to adjust the input  $(w_L, b_L, L = 1, \dots, \infty)$  and output  $(\beta_L, L = 1, \dots, \infty)$  parameters of the network. To the best of our knowledge, it is the first time that correntropy with the Gaussian kernel has been employed for training all the parameters of a constructive SLFN. C2N2 starts with zero hidden nodes. The first hidden node is added to the network. First, the input parameters  $(w_1, b_1)$  of the hidden node are calculated by employing a correntropy objective function with a Gaussian kernel. Then, they are fixed and the output parameter  $(\beta_1)$  of the node is adjusted by the correntropy objective function with Gaussian kernel. After the parameters of the first node are obtained, they are then fixed and the next hidden node is added to the network and trained. This process is iterated until the stopping condition is satisfied.

The proposed method can be viewed as an extension of CCOEN [28] with the following differences:

1. In contrast to CCOEN, which uses correntropy with a sigmoid kernel to adjust the input parameters of a cascade network, the proposed method uses correntropy with a Gaussian kernel to adjust the whole parameters of an SLFN.
2. CCOEN uses correntropy to adjust the input parameters of the new node in a cascade network to provide a more robust method. However, the output parameter of the new node in a cascade network is still adjusted based on the least mean square error. In contrast, the proposed method uses correntropy with Gaussian kernel to obtain both the input and output parameters of the new node in a constructive SLFN. Therefore, the proposed method is more robust than CCOEN and other networks introduced in [3] when the dataset is contaminated by impulsive noise.
3. Employing Gaussian kernel for correntropy as the objective function to adjust the network's parameters provides a closed-form formula introduced in the next sec-

tion. In other words, both the input and output parameters are adjusted by two closed-form formulas.

For the proposed network, each newly added node ( $L$ -th added node where  $L = 1, \dots, \infty$ ) is trained using the two phases.

In the first phase, the new node is selected from  $\mathcal{G}$ , using the following optimization problem: where

$$g_L^{\text{sim}(e)} = \operatorname{argmax}_{g_L \in \mathcal{G}} \{V(g_L)\},$$

$$V(g_L) = E \left( \frac{1}{\sqrt{2\pi}\sigma} \exp \left( -\frac{\|e_{L-1} - k_L g_L\|^2}{2\sigma^2} \right) \right).$$

From the definition of the kernel, the most similar ( $g_L^{\text{sim}(e)}$ ) activation function to the residual error of  $L - 1$  nodes network is selected from  $\mathcal{G}$  as this node is selected to maximize:

$$V(g_L) = E(\langle \Phi(e_{L-1}), \Phi(k_L g_L) \rangle),$$

$$, k_L \in \mathbb{R} - \{0\}$$

where  $\Phi$  is feature mapping. Consequently, the biggest reduction in error is obtained and the network has a more compact architecture.

In the second phase, the output parameter ( $\beta_L$ ) of the new node is adjusted whereby

$$\beta_L^{\text{sim}(e)} = \operatorname{argmax}_{\beta_L} \{V(\beta_L), \beta_L \in \mathbb{R}\},$$

$$V(\beta_L) = E \left( \frac{1}{\sqrt{2\pi}\sigma} \exp \left( -\frac{\|e_{L-1} - \beta_L g_L^{\text{sim}(e)}\|^2}{2\sigma^2} \right) \right).$$

These two phases are iterated and a new node is added in each iteration until the certain stopping condition is satisfied. This is discussed in Section 5.

After the parameters of the new node are tuned, the correntropy of the residual error (error of the network with  $L$  hidden nodes) is shown as

$$V(e_L) = E \left( \frac{1}{\sqrt{2\pi}\sigma} \exp \left( -\frac{\|e_{L-1} - \beta_L^{\text{sim}(e)} g_L^{\text{sim}(e)}\|^2}{2\sigma^2} \right) \right),$$

and the residual error is updated as follows:

$$e_L = e_{L-1} - \beta_L^{\text{sim}(e)} g_L^{\text{sim}(e)}.$$

It is important to note that this subsection only presents two optimization problems for adjusting the input and output parameters of the new node. In Section 4.4, we present a way to solve these problems.

### 4.3. Convergence Analysis

In this subsection, we prove that the correntropy of the newly constructed network undergoes a strictly increasing sequence and converges to its supremum. Furthermore, it is proven that the supremum equals the maximum. To prove the convergence of the correntropy of the network, the definitions, theorems and lemma that are presented in Section 4.1 are employed. To prove convergence of the proposed method, similarly to [3,28], we propose the following lemma and prove that the new node, which is obtained from the input side optimization problem, is not orthogonal to the residual error function.

**Lemma 2.** Given  $\text{span}(\mathcal{G})$  is dense in  $L^2(\Omega, \mathcal{F}, \mathcal{P})$  and  $e_{L-1} \in L^2(\Omega, \mathcal{F}, \mathcal{P})$ . There exists a real number  $k_L \in \mathbb{R} - \{0\}$  such that  $g_L^{\text{sim}(e)}$  is not orthogonal to  $e_{L-1}$ , where

$$g_L^{\text{sim}(e)} = \operatorname{argmax}_{g_L \in \mathcal{G}} \left( E \left( \frac{1}{\sqrt{2\pi\sigma}} \exp \left( -\frac{\|e_{L-1} - k_L g_L\|^2}{2\sigma^2} \right) \right) \right).$$

Employing Lemma 2 and what is mentioned in Section 4.1, the following theorem proves that the proposed method achieves its global solution.

**Theorem 6.** Given an SLFN with  $\tanh$  (tangent hyperbolic) function for the additive nodes, for any continuous function  $f$  and for the sequence of hidden nodes functions, obtained based on the residual error functions, i.e.,  $\{g_L^{\text{sim}(e)}\}, L \in \mathbb{N}$ , there exists a real sequence  $\{k_L; L \in \mathbb{N}\}$  such that

$$\lim_{L \rightarrow \infty} V(e_L) = V_{\max}$$

holds almost everywhere, provided that

$$\begin{aligned} g_L^{\text{sim}(e)} &= \operatorname{argmax}_{g_L \in \mathcal{G}} \{V(g_L)\}. \\ \beta_L^{\text{sim}(e)} &= \operatorname{argmax}_{\beta_L \in \mathbb{R}} \{V(\beta_L)\}, \end{aligned}$$

where

$$V_{\max} = \frac{1}{\sqrt{2\pi\sigma}}.$$

The proof of Lemma 2 and Theorem 6 contain some pure mathematics contents and are placed in Appendix A.

#### 4.4. Learning from Data Samples

In Theorem 6, we proved that the proposed network, i.e., the one hidden layer constructive neural network based on correntropy (C2N2), achieves an optimal solution. During the training process, the function form of the error is not available and the error and activation vectors are generated from the training samples. In the rest of this subsection, we propose a method to train the network from data samples.

##### 4.4.1. Input Side Optimization

The optimization problem to adjust the input parameters is as follows:

$$V_L^g = \max_{G_L} \left( E \left( \frac{1}{\sqrt{2\pi\sigma}} \exp \left( -\frac{\|E_{L-1} - k_L G_L\|^2}{2\sigma^2} \right) \right) \right).$$

On training data, expectation can be approximated as:

$$\hat{V}_L^g = \max_{G_L} \left( \frac{1}{N\sqrt{2\pi\sigma}} \sum_{i=1}^N \left( \exp \left( -\frac{\|E_{(L-1)i} - k_L G_{Li}\|^2}{2\sigma^2} \right) \right) \right).$$

The constant term  $\frac{1}{N\sqrt{2\pi\sigma}}$  can be removed and the following problem can be solved instead  $\hat{V}_L^g$ :

$$\hat{U}_L^g = \max_{G_L} \left( \sum_{i=1}^N \left( \exp \left( -\frac{\|E_{(L-1)i} - k_L G_{Li}\|^2}{2\sigma^2} \right) \right) \right).$$

Consider the following equality:

$$E_{(L-1)i} = k_L G_{Li} \quad i = 1, \dots, N.$$

In this paper, the tanh function is selected as the activation function, which is bipolar and invertible. Therefore:

$$g^{-1}\left(\frac{E_{(L-1)i}}{k_L}\right) = X_i W_{(d+1)*1} \quad i = 1, \dots, N,$$

where  $X_i = [x_i \quad 1]$ ,  $W_{(d+1)*1} = \begin{bmatrix} W_L \\ b_L \end{bmatrix}$ ,

$$X = [X_1^T, \dots, X_N^T]^T$$

in which  $abs(\cdot)$  is the absolute function. The range of  $g$  (domain of  $g^{-1}$ ) is  $[-1, 1]$ . Thus, it is necessary to rescale the error signal to be in the range. To do so,  $k_L$  is assigned as follows:

$$k_L = \frac{\max(abs(E_L))}{\lambda'}$$

where  $\lambda' \in (-1, 1) - \{0\}$ . Let  $H_{Li} = g^{-1}\left(\frac{E_{(L-1)i}}{k_L}\right)$ , and therefore, the term

$$\left\| E_{(L-1)i} - k_L G_{Li} \right\|^2$$

can be replaced by

$$\|H_{Li} - X_i W_L\|^2,$$

and thus the following problem is presented to adjust the input parameters

$$\hat{U}_L^g = \max_{W_L} \left( \sum_{i=1}^N \left( \exp \left( -\frac{\|H_{Li} - X_i W_L\|^2}{2\sigma^2} \right) \right) \right).$$

To achieve better generalization performance, the norm of the weights needs to be kept minimized too; thus, the problem above is reformulated as

$$\hat{U}_L^g = \max_{W_L} \left( \sum_{i=1}^N \left( \exp \left( -\frac{\|H_{Li} - X_i W_L\|^2}{2\sigma^2} \right) \right) - \frac{C}{2} \|W_L\|^2 \right).$$

It should be considered that if  $C = 0$ , both problems from above are equivalent. Since the necessary condition for convergence is that in each step and by adding each node amount of correntropy of the error should be increased, in the experiment section,  $C \approx 0$  and other amounts for  $C$  are checked and the best result is selected. This guarantees convergence of the method according to Theorem 6.

The half-quadratic method is employed to adjust the input parameters. Based on Proposition 1, we have

$$\square_L^g(\alpha, W_L) = \max_{W_L, \alpha} \left( \sum_{i=1}^N \left( \alpha_i \frac{\|H_{Li} - X_i W_L\|^2}{2\sigma^2} - \Phi(\alpha_i) \right) - \frac{C}{2} \|W_L\|^2 \right).$$

The local solution of the above optimization problem is adjusted using the following iterative process:

$$\left\{ \begin{array}{l} \alpha_i^{t+1} = -G(H_{Li} - X_i W_L) \\ W_L^{t+1} = \arg_{W_L} \max \left( \sum_{i=1}^N \left( \alpha_i^{t+1} \frac{\|H_{Li} - X_i W_L\|^2}{2\sigma^2} \right) - \frac{C}{2} \|W_L\|^2 \right) \end{array} \right. ,$$

i.e., the following optimization problem needs to be solved in each iterate:

$$\mathcal{V}_L^g(\alpha, W_L) = \max_{W_L} \left( \sum_{i=1}^N \left( \alpha_i^{t+1} \frac{\|H_{Li} - X_i W_L\|^2}{2\sigma^2} \right) - \frac{C}{2} \|W_L\|^2 \right).$$

Since  $\sigma^2$  is a constant term, it can be removed from the optimization problem. Then, the optimization problem can be multiplied by  $\frac{1}{C}$ . We set  $C' = \frac{1}{C}$ . Thus, the following constraint optimization problem is obtained:

$$\begin{cases} \max \sum_{i=1}^N \frac{C'}{2} (\alpha_i^{t+1} \zeta_i^2) - \frac{1}{2} \|W_L\|^2 \\ \text{s.t. } X_i W_L = H_{Li} - \zeta_i \quad i = 1, \dots, N \end{cases}$$

The Lagrangian is constituted as

$$L(\zeta_i, \eta_i, W_L) = \sum_{i=1}^N \frac{C'}{2} (\alpha_i^{t+1} \zeta_i^2) - \|W_L\|^2 - \sum_{i=1}^N \eta_i (X_i W_L - H_{Li} + \zeta_i).$$

The derivations of the Lagrangian function with respect to its variables are the following

$$\frac{\partial L}{\partial W_L} = 0 \rightarrow W_L = - \sum_{i=1}^N \eta_i X_i^T = -X^T \eta,$$

where  $\eta = [\eta_1, \dots, \eta_N]^T$ .

$$\frac{\partial L}{\partial \zeta_i} = 0 \rightarrow \eta_i = C' \alpha_i^{t+1} \zeta_i \quad i = 1, \dots, N,$$

$$\frac{\partial L}{\partial \eta_i} = 0 \rightarrow X_i W_L - H_{Li} + \zeta_i = 0 \quad i = 1, \dots, N.$$

Now we consider two cases.

Case 1.  $d \leq N$

By substituting derivatives in

$$W_L = - \sum_{i=1}^N C' \alpha_i^{t+1} \zeta_i X_i^T$$

we obtain

$$\begin{aligned} W_L &= - \sum_{i=1}^N C' \alpha_i^{t+1} (-X_i W_L + H_{Li}) X_i^T \\ &= \sum_{i=1}^N C' \alpha_i^{t+1} (X_i W_L) X_i^T - \sum_{i=1}^N C' \alpha_i^{t+1} H_{Li} X_i^T. \end{aligned}$$

Let  $\Psi$  be a diagonal matrix with  $\Psi_{ii} = \alpha_i$ ; therefore,

$$W_L - (C' X^T \Psi X W_L) = -C' X^T \Psi H_L$$

$$W_L (I - C' X^T \Psi X) = -C' X^T \Psi H_L$$

$$W_L = \left( X^T \Psi X - \frac{I}{C'} \right)^{-1} X^T \Psi H_L.$$

Case 2.  $d \geq N$

By substituting derivatives in

$$X W_L - H_L + \zeta = 0 \rightarrow -X X^T \eta - H_L + \zeta = 0,$$

$$\eta = C' \Psi \zeta.$$

we obtain

$$\begin{aligned}
 & -XX^T C' \Psi \zeta - H_L + \zeta = 0 \\
 & \rightarrow -XX^T C' \Psi \zeta + \zeta = H_L \\
 & \rightarrow (-C' XX^T \Psi + I) \zeta = H_L \\
 & \rightarrow \left(-XX^T \Psi + \frac{I}{C'}\right) C' \zeta = H_L \\
 & \rightarrow C' \zeta = \left(-XX^T \Psi + \frac{I}{C'}\right)^{-1} H_L
 \end{aligned}$$

Then

$$\begin{aligned}
 & -\left(X^T\right)^+ W_L = \Psi \left(-XX^T \Psi + \frac{I}{C'}\right)^{-1} H_L \\
 & \rightarrow W_L = X^T \Psi \left(XX^T \Psi - \frac{I}{C'}\right)^{-1} H_L
 \end{aligned}$$

Thus, the input parameters are obtained by the following iterative process:

$$\left\{ \begin{array}{l} \alpha_i^{t+1} = -G(H_{Li} - X_i W_L^t) \\ W_L^{t+1} = \left(X^T \Psi^t X - \frac{I}{C'}\right)^{-1} X^T \Psi^t H_L \\ \text{or} \\ W_L^{t+1} = X^T \Psi^t \left(XX^T \Psi^t - \frac{I}{C'}\right)^{-1} H_L \end{array} \right.$$

#### 4.4.2. Output Side Optimization

When the input parameters of the new node are obtained from the previous step, the new node is named  $G_L^{\text{sim}(e)} = \{G_{L1}^{\text{sim}(e)}, \dots, G_{LN}^{\text{sim}(e)}\}$ , where  $G_{Li}^{\text{sim}(e)} = g_L^{\text{sim}(e)}(x_i)$  and the output parameter is adjusted using the following optimization problem:

$$V_L^\beta = \max_{\beta_L} \left( E \left( \frac{1}{\sqrt{2\pi\sigma}} \exp \left( -\frac{\|e_{L-1} - \beta_L g_L^{\text{sim}(e)}\|^2}{2\sigma^2} \right) \right) \right).$$

The expectation can be approximated on training samples:

$$\hat{V}_L^\beta = \max_{\beta_L} \left( \frac{1}{N\sqrt{2\pi\sigma}} \sum_{i=1}^N \left( \exp \left( -\frac{\|E_{(L-1)i} - \beta_L G_{Li}^{\text{sim}(e)}\|^2}{2\sigma^2} \right) \right) \right).$$

The constant term  $\frac{1}{N\sqrt{2\pi\sigma}}$  can be removed and the following problem can be solved instead  $\hat{V}_L^\beta$ :

$$\hat{U}_L^\beta = \max_{\beta_L} \left( \sum_{i=1}^N \left( \exp \left( -\frac{\|E_{(L-1)i} - \beta_L G_{Li}^{\text{sim}(e)}\|^2}{2\sigma^2} \right) \right) \right).$$

Similar to the previous step, the half-quadratic method is employed to adjust the output parameter. Based on Proposition 1, we obtain

$$u_L^\beta(\gamma, \beta_L) = \max_{\beta_L, \gamma} \left( \sum_{i=1}^N \left( \gamma_i \frac{\|E_{(L-1)i} - \beta_L G_{Li}^{\text{sim}(e)}\|^2}{2\sigma^2} - \Phi(\gamma_i) \right) \right).$$

The local solution of the above optimization problem is adjusted using the following iterative process:

$$\begin{cases} \gamma_i^{t+1} = -k_\sigma (E_{Li} - \beta_L^t G_{Li}^{\text{sim}(e)}) \\ \beta_L^{t+1} = \arg_{\beta_L} \max \left( \sum_{i=1}^N \left( \gamma_i^{t+1} \frac{\|E_{Li} - \beta_L^t G_{Li}^{\text{sim}(e)}\|^2}{2\sigma^2} \right) \right) \end{cases} .$$

i.e., the following optimization problem is required to be solved in each iteration:

$$\begin{aligned} \mathcal{V}_L^\beta(\gamma, \beta_L) &= \max_{\beta_L} \left( \sum_{i=1}^N \left( \gamma_i^{t+1} \frac{\|E_{Li} - \beta_L G_{Li}^{\text{sim}(e)}\|^2}{2\sigma^2} \right) \right), \\ \mathcal{V}_L^\beta(\gamma, \beta_L) &= \max_{\beta_L} \left( \frac{1}{2\sigma^2} \left( (E_L - \beta_L G_L^{\text{sim}(e)}) \Theta (E_L - \beta_L G_L^{\text{sim}(e)})^T \right) \right). \end{aligned}$$

where  $\Theta$  is a diagonal matrix with  $\Theta_{ii} = \gamma_i, i = 1, \dots, N$ .

$$\mathcal{V}_L^\beta(\gamma, \beta_L) = \max \left( \frac{1}{2\sigma^2} \left( E_L \Theta E_L^T + \beta_L^2 G_L^{\text{sim}(e)} \Theta G_L^{\text{sim}(e)T} - \beta_L E_L \Theta G_L^{\text{sim}(e)T} - \beta_L G_L^{\text{sim}(e)} \Theta E_L^T \right) \right).$$

The optimum output weight is adjusted by differentiating  $\mathcal{V}_L^\beta(\gamma, \beta_L)$  with respect to  $\beta_L$  as

$$\begin{aligned} (2\beta_L G_L^{\text{sim}(e)} \Theta G_L^{\text{sim}(e)T} - E_L \Theta G_L^{\text{sim}(e)T} - G_L^{\text{sim}(e)} \Theta E_L^T) &= 0, \\ \beta_L &= \frac{E_L \Theta G_L^{\text{sim}(e)T}}{G_L^{\text{sim}(e)} \Theta G_L^{\text{sim}(e)T}}. \end{aligned}$$

Finally, the output weight is adjusted by the following iterative process:

$$\begin{cases} \gamma_i^{t+1} = -k_\sigma (E_{Li} - \beta_L^t G_{Li}^{\text{sim}(e)}) \\ \beta_L^{t+1} = \frac{E_L \Theta G_L^{\text{sim}(e)T}}{G_L^{\text{sim}(e)} \Theta G_L^{\text{sim}(e)T}} \end{cases} .$$

In these two phases, the parameters of the new node ( $L$ -th added node where  $L \in \mathbb{N}$ ) are tuned and then fixed. This process is iterated for each new node until the predefined condition is satisfied. The following proposition demonstrates that for each node, the algorithm converges.

**Proposition 2.** The sequences  $\{\square_L^g(\alpha^t, W_L^t), t = 1, 2, \dots\}$  and  $\{\mathcal{U}_L^\beta(\gamma^t, \beta_L^t), t = 1, 2, \dots\}$  converge.

**Proof.** From Theorem 5 and Proposition 1, we have  $u_L^\beta(\gamma^t, \beta_L^t) \leq u_L^\beta(\gamma^{t+1}, \beta_L^t) \leq u_L^\beta(\gamma^{t+1}, \beta_L^{t+1})$  and  $(\square_L^g(\alpha^t, W_L^t) \leq u_L^g(\alpha^{t+1}, W_L^t) \leq u_L^g(\alpha^{t+1}, W_L^{t+1}))$ . Thus, the non-decreasing sequence  $\{u_L^\beta(\gamma^t, \beta_L^t), t = 1, 2, \dots\}$  ( $\{u_L^g(\alpha^t, W_L^t), t = 1, 2, \dots\}$ ) converges since the correntropy is upper bounded.  $\square$

**Proposition 3.** When  $\Theta = I$ , the output weight that is adjusted by the correntropy criterion is equivalent to the output weight that is adjusted by the MSE-based method such as IELM.

**Proof.** Suppose that  $\Theta = I$ , by  $\beta_L = \frac{E_b \theta \mathbf{G}_L^{\text{sim}(e)T}}{\mathbf{G}_L^{\text{sim}(e)} \Theta \mathbf{G}_L^{\text{sim}(e)T}}$ , we have

$$\beta_L = \frac{E_L \mathbf{G}_L^{\text{sim}(e)T}}{\mathbf{G}_L^{\text{sim}(e)} \mathbf{G}_L^{\text{sim}(e)T}}.$$

□

The training process of the proposed method is summarized in the following Algorithm 1 (C2N2).

---

#### Algorithm 1 C2N2

---

**Input:** training samples  $\chi = \{x_i, y_i\}_{i=1}^N$

**Output:** Optimal input and output weights  $\beta_L, W_L, L = 1, \dots, L_{max}$

*Initialization:* Maximum number of hidden nodes  $L_{max}$ , regularization term  $C'$ , maximum input side and output side iterations  $IT1, IT2$ , error  $E_0 = [y_1, \dots, y_N]$ .

**For**  $L = 1 : L_{max}$

    Step 1: Calculate  $H_L$  and  $X$

**For**  $k = 1 : IT1$

            Update input parameters

**End**

    Step 2: Calculate the hidden node vector,  $\mathbf{G}_L^{\text{sim}(e)}$  by previous step

**For**  $k = 1 : IT2$

            Update output weight

**End**

    Update error as  $E_L = E_{L-1} - \beta_L^{\text{sim}(e)} \mathbf{G}_L^{\text{sim}(e)}$

**End**

---

**Remark 1.** The auxiliary variables  $\gamma_i$  and  $\alpha_i, i = 1, \dots, N$  are utilized to reduce the effect of noisy data. For the samples with a high amount of error, these variables are very small; thus, these samples have slight effects on the optimization of the parameters of the network, which results in a more robust network.

## 5. Experimental Results

This section compares C2N2 with RBF, CCN and other constructive networks that are presented in [3]. The networks, whose hidden nodes' input parameters are trained by the objective functions  $V_1, V_2, V_3, \sqrt{V_1}, \sqrt{V_2}, \sqrt{V_3}$  that are introduced in [3], are denoted by  $N_1, \dots, N_6$ . In addition to the mentioned methods, the proposed method is compared to the state-of-the-art constructive networks such as the orthogonal least square cascade network (OLSCN) [4] and the one hidden layer constructive network (OHLCN) introduced in [5]. Moreover, C2N2 is also compared with state-of-the-art robust learning methods including Multi-Layer Perceptron based on MCC (MLPMCC) [17], Multi-Layer Perceptron based on Minimum Error Entropy (MLPMEE) [17] and Robust Least Square Support Vector Machine (RLS-SVM) [35] and the recent work, CCOEN [28].

The rest of this section is organized as follows. Section 5.1 describes a framework for the experiments. The presented theorem and hyperparameters ( $L, C'$  and  $\eta'$ ) are investigated in Section 5.2. In Section 5.3, the presented method is compared to N1-N6, CCN, RBF and some state-of-the-art constructive networks including OHLCN and OLSCN. Experiments are performed on several synthetic and benchmark datasets that are contaminated with impulsive noise (one of the most popular types of non-Gaussian noise). In this part, experiments are also performed in the absence of impulsive noise. Section 5.4 compares the proposed method with state-of-the-art robust learning methods including MLPMEE, MLPMCC, RLS-SVM and CCOEN on various types of datasets.

### 5.1. Framework for Experiments

This part presents a framework for the experiments. The framework includes the type of activation function for C2N2 and other mentioned methods, type of kernel, kernel parameters ( $\sigma$ ), range of hyperparameters ( $L, C', \lambda'$ ) and dataset specification.

#### 5.1.1. Activation Function and Kernel

For the proposed method, the tangent hyperbolic activation function is used. It is represented as follows (see Figure 3):

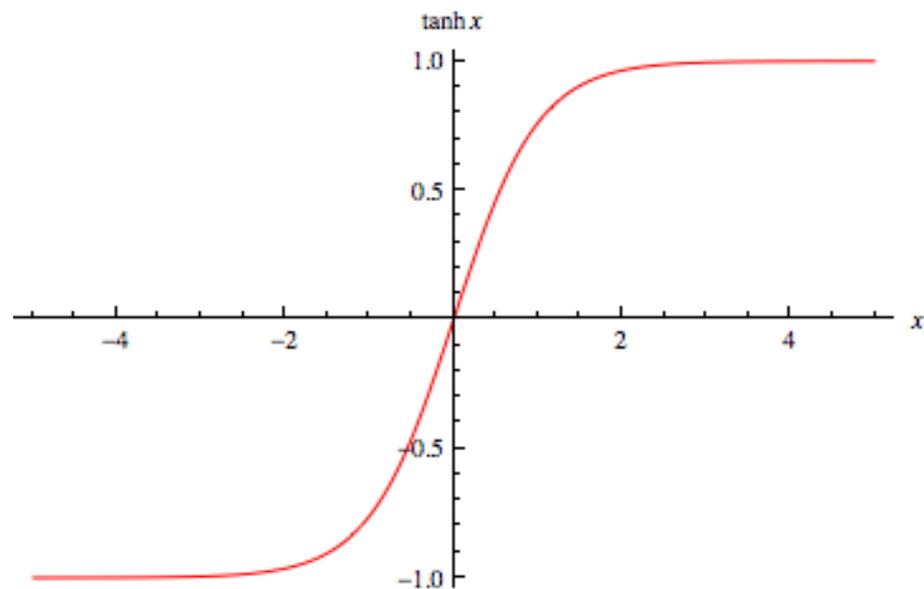


Figure 3. Tangent hyperbolic function.

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

For networks  $N_1 - N_6$ , CCN, OLSCN, OHLCN, MLP MCC and MLP MEE, the sigmoid activation function is used. This function is represented and displayed as follows (see Figure 4):

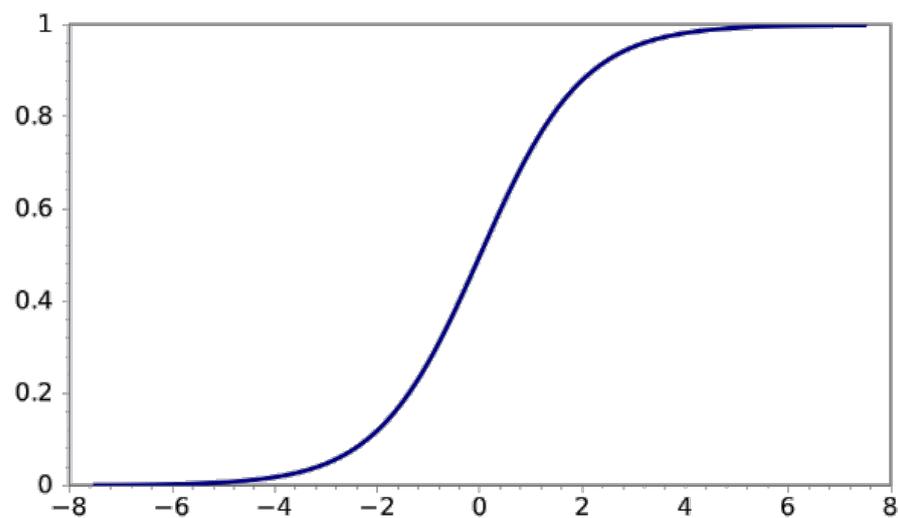


Figure 4. Sigmoid function.

For the proposed method and RLS-SVM, RBF kernel is used. It is shown as

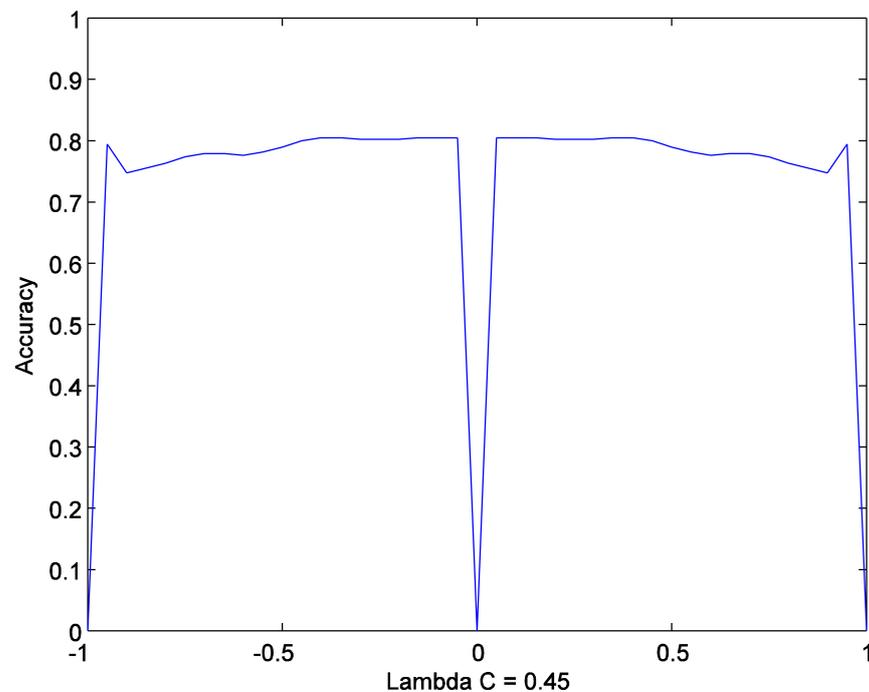
$$K(X, Y) = \exp\left(-\frac{\|X - Y\|^2}{2\sigma^2}\right)$$

In the experiments, the optimum kernel parameter ( $\sigma$ ) is selected from the set

$$\{0.1, 0.5, 1, 10, 15\}.$$

### 5.1.2. Hyperparameters

The method has three hyperparameters. These parameters help to avoid over or underfitting, which improves performance. The first parameter is a number of hidden nodes ( $L$ ). The optimum number of hidden nodes is selected from the set  $\{1, \dots, 8\}$ . Due to the boundedness of tanh function ( $-1 < \tanh(x) < 1$ ), the error signal must be scaled to this range. Thus,  $\lambda'$  should be selected from the set  $\{-0.9, \dots, -0.1, 0.1, 0.2, \dots, 0.9\}$ . Figure 5 shows that accuracy is symmetric with respect to  $\lambda'$ . Thus,  $\lambda'$  should be selected from the set  $\{0.1, \dots, 0.9\}$ . The possible range for  $C'$  is investigated in the next part.



**Figure 5.** Effect of  $\lambda'$  on accuracy. Parameter  $C'$  is set to 0.45. The experiment is performed on the diabetes dataset with the network with only one hidden node.

### 5.1.3. Data Normalization

In this paper, the input vector of data samples is normalized into the range  $[-1, 1]$ . For regression datasets, their targets are normalized into the range  $[0, 1]$ .

In this paper, most of the datasets are taken from the UCI Machine Learning Repository [36] and Statlib [37]. These datasets are specified in Tables 1 and 2.

**Table 1.** Specification of the regression problem.

Datasets	#Train	#Test	#Features
Basketball	64	32	4
Strike	416	209	6
Bodyfat	168	84	14
Quake	1452	726	3
Autoprice	106	53	9
Baloon	1334	667	2
Pyrim	49	25	27
Housing	337	169	13
Abalone	836	3341	8
Cleveland	149	148	13
Cloud	54	54	7

**Table 2.** Specification of the classification problem.

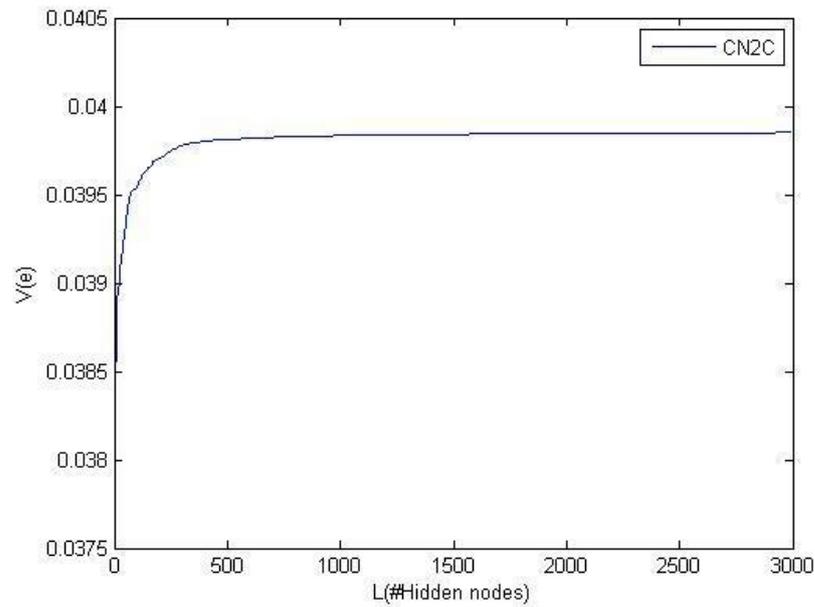
Dataset	#Train	#Test	#Features
Ionosphere	175	176	34
Australian Credit	460	230	6
Diabetes	512	256	8
Colon	32	30	2000
Liver	230	115	6
Leukemia	36	36	7129
Dimdata	1000	3192	14

## 5.2. Convergence

This part investigates the convergence of the proposed method (Theorem 6), followed by an investigation of the hyperparameters.

### 5.2.1. Investigation of Theorem 6

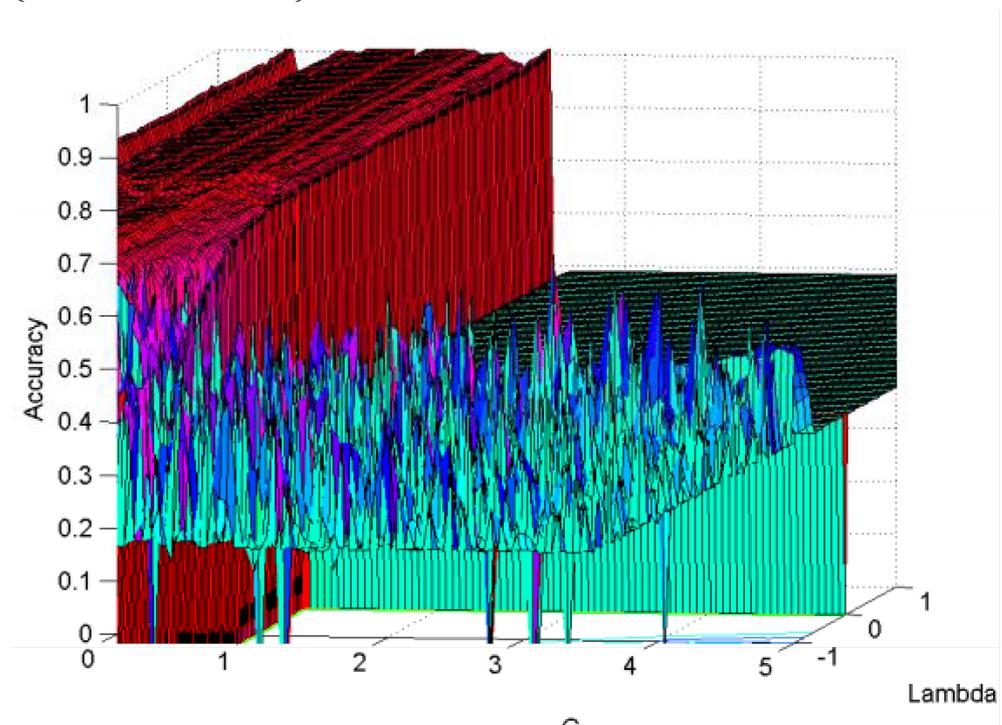
The main goal of this paper is to maximize the correntropy of the error function. Regarding the kernel definition and due to maximization of correntropy, the approximator (output of the neural network,  $f_L$ ) has the most similarity to the target function ( $f$ ). Theorem 6 proves that the proposed method obtains the optimal solution, i.e., the correntropy of the error function is maximized. This part investigates the convergence of the proposed method. In this experiment, the kernel parameter is set to 10; thus, the optimum value for correntropy is  $v(e = 0) = V^{\max} = 0.0399$ . Figure 6 shows the convergence of C2N2 to the optimum value in the approximation of the sinc function.



**Figure 6.** Convergence of the proposed method in the approximation of Sinc function when  $\sigma = 10$ . It converges to  $V^{\max} = 0.0399$ .

### 5.2.2. Hyperparameter Evaluation

To evaluate parameters  $C'$  and  $\lambda'$ , C2N2 with only one hidden node was experimented on using the diabetes dataset. Figure 7 shows that the best amount for parameter  $C'$  is in the range  $[0, 1]$ . Thus, in the experiments, parameter  $C'$  is selected from the set  $\{0.05, 0.15, 0.25, \dots, 0.95\}$ .



**Figure 7.** Effect of hyperparameters  $(C', \lambda')$  on accuracy.

### 5.3. Comparison

This part compares the proposed method with the networks  $N_1, \dots, N_6$ , CCN, OHLCN, OLSCN, and RBF in the presence and absence of non-Gaussian noise. One of the worst types of non-Gaussian noise is impulsive noise.

For  $\lambda' = 0$ , the accuracy is set to zero. This type of noise adversely affects the performance of MSE-based methods such as the networks  $N_1 - N_6$ , CCN, OHLCN and OLSCN. In this part and part D, we perform experiments similar to [4]. We calculated the RMSE (classification accuracy) on the testing dataset after each hidden unit was added and reported the lowest (highest) RMSE (accuracy) along with the corresponding network size. Similar to [4], experiments were carried out in 20 trials and the results (RMSE (accuracy) and number of nodes) averaged over 20 trials are listed in Tables 3–6.

In all result tables, the best results are shown in bold and underlined. The results that are close to the best ones are in bold.

**Table 3.** Performance comparison of C2N2 and the networks  $N_4, N_5, N_6, CCN$  and RBF: benchmark regression dataset.

Datasets	C2N2			$N_4$			$N_5$			$N_6$			CCN			RBF		
	Testing RMSE	#N	Time (s)	Testing RMSE	#N	Time (s)	Testing RMSE	#N	Time (s)	Testing RMSE	#N	Time (s)	Testing RMSE	#N	Time (s)	Testing RMSE	#N	Time (s)
Autoprice	<u>0.2689</u>	4.8	0.54	0.2996	2	1.49	<u>0.2689</u>	9.2	3.99	0.3681	5	6.99	0.2758	<u>1.5</u>	1.67	0.2725	79	0.008
Autoprice (Noise)	<u>0.2770</u>	6.7	0.70	0.5521	<b>1.33</b>	1.07	0.3610	2.60	0.43	0.4295	3.77	1.41	0.4768	<u>1.2</u>	1.89	0.9082	79	0.009
Baloon	0.1065	5.9	10.73	0.1163	5.75	5.21	0.1066	10	4.96	0.1257	8.3	9.97	0.1317	<b>3.58</b>	2.09	<u>0.0563</u>	150	0.086
Baloon (Noise)	<b>0.1056</b>	5.1	3.25	0.1166	<u>2</u>	1.32	0.1252	8.9	4.66	0.1281	5	3.31	0.1358	2.9	2.12	<u>0.1051</u>	1501	0.1351
Pyrim	<b>0.0482</b>	6.7	0.23	0.0843	<u>1</u>	0.27	0.2062	<u>1.2</u>	0.07	0.1696	<u>1</u>	0.17	0.1694	<u>1</u>	0.17	0.0842	37	0.0090
Pyrim (noise)	<u>0.0521</u>	4.9	0.16	0.6666	<u>1.5</u>	0.32	0.4150	<u>1</u>	0.036	0.6203	<u>1</u>	0.62	0.5712	<u>1</u>	0.62	1.5034	37	0.0043

**Table 4.** Performance comparison of RBF,  $N_4, N_5, N_6, CCN$  and C2N2: classification datasets.

Datasets	$N_4$			$N_5$			$N_6$			RBF			CCN			C2N2		
	Testing Rate (%)	#N	Time (s)	Testing Rate (%)	#N	Time (s)	Testing Rate (%)	#N	Time (s)	Testing Rate (%)	#N	Time (s)	Testing Rate (%)	#N	Time (s)	Testing Rate (%)	#N	Time (s)
Ionospher	70.97	<b>1.10</b>	0.12	65.45	<u>1</u>	0.11	78.98	2.40	0.25	82.61	175	0.14	78.04	<u>1.60</u>	0.21	<u>86.70</u>	1.30	0.22
Colon	62.00	<u>1</u>	0.02	63.00	<b>1.15</b>	0.03	62.33	<b>1.35</b>	0.38	90.50	32	0.12	64.04	<u>1.05</u>	0.29	<u>91.50</u>	<u>1</u>	0.15
Leukemia	64.44	<u>1</u>	0.03	64.44	<u>1</u>	0.05	72.44	<u>1</u>	0.04	88.61	36	0.22	83.71	<u>1.3</u>	0.31	94.72	1	0.20
Dimdata	89.74	7.05	15.31	88.44	6.60	9.39	88.77	4.30	7.34	<u>95.05</u>	1000	4.36	88.37	<u>3.60</u>	8.98	93.73	3.50	8.29

**Table 5.** Performance comparison of C2N2 and the state-of-the-art constructive networks OLSCN and OHLCN: benchmark regression dataset.

Dataset	C2N2			OLSCN			OHLCN		
	Testing RMSE	#Nodes	Time (s)	Testing RMSE	#Nodes	Time (s)	Testing RMSE	#Nodes	Time (s)
Housing	<u>0.0966</u>	5.6	0.79	0.0988	<u>1.9</u>	1.44	0.0993	2.8	0.38
Housing (Noise)	<u>0.0978</u>	5.87	1.07	0.2411	<u>1.1</u>	1.53	0.1824	4.3	0.09

Table 5. Cont.

Dataset	C2N2			OLSCN			OHLCN		
	Testing RMSE	#Nodes	Time (s)	Testing RMSE	#Nodes	Time (s)	Testing RMSE	#Nodes	Time (s)
Strike	<b>0.2807</b>	2.8	1.11	<b>0.2818</b>	<u>2</u>	2.77	0.2888	3.3	0.62
Strike (Noise)	<u>0.2817</u>	4.4	0.87	0.3017	<u>2</u>	2.21	0.3912	6.0	0.07
Quake	<b>0.1784</b>	6.6	2.05	0.1821	<u>2</u>	3.88	0.1815	6	0.023
Quake (noise)	<u>0.1744</u>	2	0.42	0.1870	<u>1.75</u>	2.21	0.1849	4	0.021

Table 6. Performance comparison of C2N2 AND the state-of-the-art constructive networks OLSCN and OHLCN: benchmark classification dataset.

Dataset	C2N2			OLSCN			OHLCN		
	Testing RMSE	#Nodes	Time (s)	Testing RMSE	#Nodes	Time (s)	Testing RMSE	#Nodes	Time (s)
Australian (Noise)	<u>86.77</u>	3	0.48	<b>86.67</b>	2	10.88	<b>86.09</b>	<u>1</u>	1.01
Liver (Noise)	<u>65.70</u>	4	0.25	<b>65.12</b>	<u>2</u>	1.01	53.49	9	0.02
Diabete (noise)	<b>79.95</b>	<u>1</u>	1.11	78.65	2	3.36	40.56	<u>1</u>	6.23

### 5.3.1. Synthetic Dataset (Sinc Function)

Figure 8 compares C2N2 with RBF and the network N1 in the approximation of the Sinc function. In Figure 9, the experiment is performed in the presence of impulsive noise.

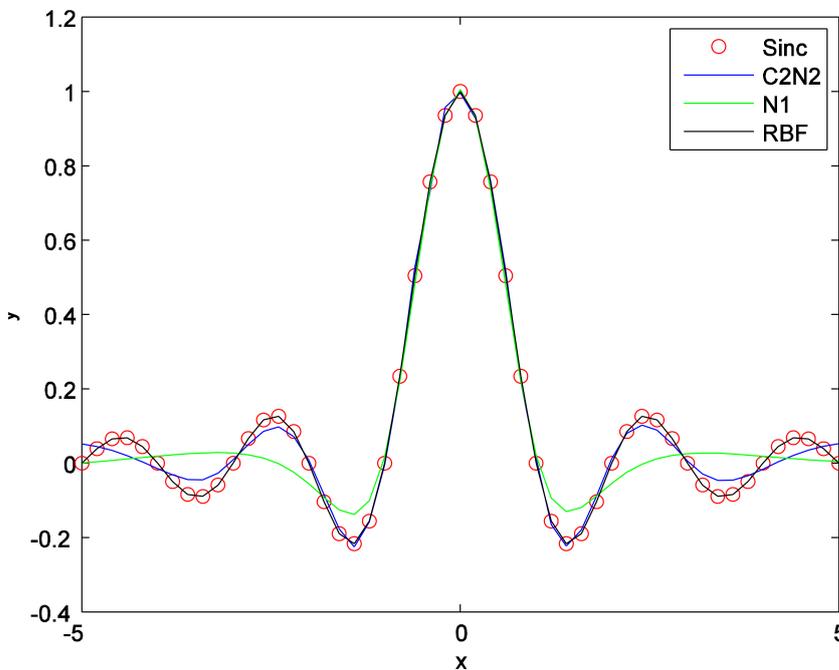


Figure 8. Comparison of C2N2 with RBF and the network  $N_1$ . The experiment is performed on the approximation of the Sinc function.

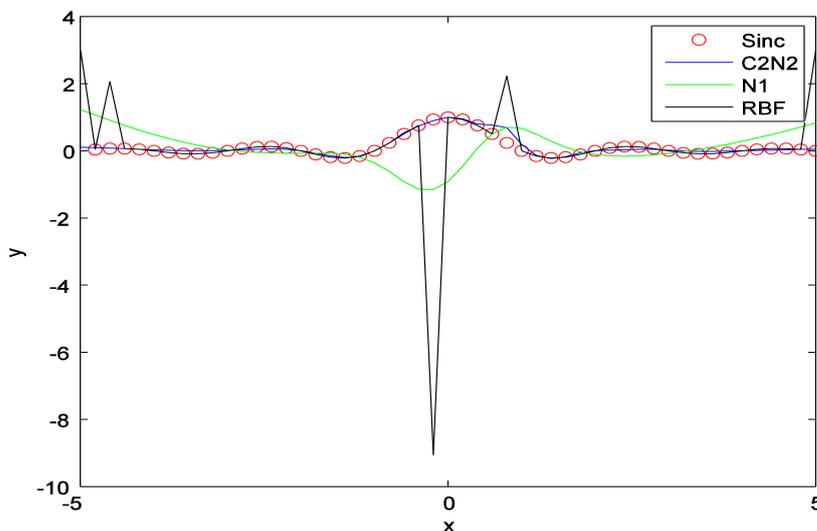


Figure 9. Comparison of C2N2 with RBF and the network  $N_1$ . The experiment is performed on the approximation of the Sinc function and in the presence of impulsive noise.

### 5.3.2. Other Synthetic Dataset

The following regression problems are used to evaluate the performance of the proposed method in comparison to the networks  $N_1, N_2$  and  $N_3$ .

$$f^{(1)}(x_1, x_2) = 10.391((x_1 - 0.4)(x_2 - 0.6) + 0.36)$$

$$f^{(2)}(x_1, x_2) = 24.234r^2(0.75 - r^2)$$

where

$$\begin{aligned}
 r &= (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \\
 f^{(3)}(x_1, x_2) &= 42.659 \left( 0.1 + r_1 \left( 0.05 + r_1^4 - 10r_1^2 r_2^2 + 5r_2^4 \right) \right) \\
 r_1 &= x_1 - 0.5 \text{ and } r_2 = x_2 - 0 \\
 f^{(4)}(x_1, x_2) &= \\
 &1.3365 \left( 1.5(1 - x_1) + e^{2x_1 - 1} \sin \left( 3\pi(x_1 - 0.6)^2 \right) + e^3(x_2 - 0.5) + \sin \left( 4\pi(x_2 - 0.9)^2 \right) \right)
 \end{aligned}$$

For each of the above functions, 225 pairs  $(x_1, x_2)$  are generated randomly in the interval  $[0, 1]$ . For each piece of training data, its target is assigned as

$$\begin{aligned}
 y_{ji} &= f_l^{(j)}(x_{i1}, x_{i2}) = f^{(j)}(x_{i1}, x_{i2}) + \eta_{il} \\
 i &= 1, \dots, 225 \text{ and } j = 1, \dots, 4 \text{ and } l = 0, 1
 \end{aligned}$$

where  $\eta_i$  is noise that is added to the target of the data samples. In this section, the index of the noise ( $l$ ) is:

$$\begin{cases} 0. \text{ without noise} \\ 1. \eta_{il} = \text{impulse noise} \end{cases}$$

For impulse noise, the outputs of five data samples are changed by extra high values using the uniform distribution.

#### 5.4. Discussion

##### 5.4.1. Discussion on Table 7

Table 7 compares C2N2 with the networks  $N_1, N_2$  and  $N_3$ . From the table, we can see that in the absence of noise, the proposed method outperforms the other methods on the datasets  $f_0^{(2)}, f_0^{(3)}$  and  $f_0^{(4)}$ . For the dataset  $f_0^{(1)}$ , the best result is for the network  $N_2$ . We added impulsive noise to the dataset and again performed experiments. From Table 7, we can see that the proposed method is more stable when data are contaminated with impulsive noise. For example, for the dataset  $f_0^{(2)}$ , the RMSEs of C2N2 and  $N_1$  to  $N_3$  are close. However, in the presence of impulsive noise for the dataset  $f_1^{(2)}$ , the RMSEs for C2N2 and  $N_1$  to  $N_3$  are 0.2487, 0.3676, 0.2790, and 0.3226 respectively. This means that noisy data samples have less effect on the proposed method in comparison to the other mentioned methods and the proposed method is more stable. The goal of any learning method is to increase performance. Thus, in this paper, we focus on RMSE. However, from the architecture viewpoint, the proposed method tends to have a smaller number of nodes in 50% of the datasets.

##### 5.4.2. Why C2N2 Denies Impulse Noises?

Regarding optimization problems, for noisy (impulsive noise) data, auxiliary variables  $\alpha_i^t, \gamma_i^t$  are low. Thus, such data have a small role in optimization problems; parameters of the new node are obtained based on noise-free data (Remark 1). Table 8 shows the auxiliary variables for several noisy and noise-free data samples.

##### 5.4.3. Benchmark Dataset

In this part, several regression and classification datasets are contaminated with impulse noise. At this time, as in [38], we produce impulsive noise by generating random real numbers from the following distribution function, and then we add them to data samples:

$$\eta = 0.95N(0, 10^{-4}) + 0.05N(0, 10)$$

where  $N(\mu, \sigma)$  is a Gaussian distribution function with the mean  $\mu$  and variance  $\sigma$ . For the regression dataset, we add noise to its target. For the classification dataset, we add noise

to its input feature vector. Experiments on these datasets confirm the robustness of the proposed method in comparison with  $N_4, \dots, N_6$ , CCN and RBF, OLSCN and OHLCN.

**Table 7.** Performance comparison of C2N2 and the networks  $N_1, N_2$  and  $N_3$ : synthetic regression dataset.

Datasets	C2N2			$N_1$			$N_2$			$N_3$		
	Testing RMSE	#N	Time (s)	Testing RMSE	#N	Time (s)	Testing RMSE	#N	Time (s)	Testing RMSE	#N	Time (s)
$f_0^{(1)}$	0.1358	<u>3.70</u>	0.69	0.1555	4.85	1.48	<u>0.1284</u>	6.30	1.78	0.1536	7.05	0.81
$f_1^{(1)}$	<u>0.1587</u>	6.25	0.91	0.3244	<b>3.70</b>	0.44	0.2460	<u>3.20</u>	0.72	0.2750	4.40	2.40
$f_0^{(2)}$	<b>0.1963</b>	<u>2.70</u>	1.09	<b>0.1953</b>	5.90	1.91	<b>0.1953</b>	4.55	1.57	<b>0.1954</b>	5.45	2.18
$f_1^{(2)}$	<u>0.2487</u>	<b>3.55</b>	3.98	0.3676	5.25	1.08	0.2790	<u>3.25</u>	0.89	0.3226	5.65	1.16
$f_0^{(3)}$	<u>0.0892</u>	<u>3.15</u>	1.05	0.0940	6.80	0.12	0.0943	<b>3.30</b>	1.34	0.0935	6.40	0.34
$f_1^{(3)}$	<u>0.1416</u>	4.15	1.12	0.2949	4.15	0.64	0.2309	<u>2.35</u>	0.51	0.2555	4.35	0.57
$f_0^{(4)}$	<b>0.1136</b>	<u>2.10</u>	1.81	<b>0.1141</b>	4.55	1.34	<b>0.1139</b>	8.10	2.02	0.1963	4.70	0.87
$f_1^{(4)}$	<u>0.1360</u>	7.70	3.01	0.2928	<u>1.65</u>	1.54	0.2569	4.20	0.98	0.2961	4.85	0.45

**Table 8.** Amount of auxiliary variables for noisy and noise-free data. The experiment is performed on the  $f^1$  dataset.

Noisy Data	#3	#10	#36	#55	#100
$(\alpha_i, \gamma_i)$	$(-0.2351, -3.29 * 10^{-22})$	$(-0.2551, -4.3 * 10^{-22})$	$(-0.2310, -1.04 * 10^{-24})$	$(-0.2607, -8.06 * 10^{-22})$	$(-0.2482, -6.12 * 10^{-23})$
Noise-free data	#4	#11	#37	#56	#101
$\alpha_i, \gamma_i$	$(0.3837, -0.3989)$	$(-0.3983, -0.3989)$	$(-0.3989, -0.3955)$	$(-0.3989, -0.3982)$	$(-0.3988, -0.3988)$

5.4.4. Discussion on Table 3

Table 3 compares C2N2 with the networks  $N_4, N_5, N_6$ , CCN and RBF on the Autoprice, Baloon and Pyrim datasets in the presence and absence of impulsive noise. It shows that C2N2 is more stable than the networks  $N_4$  to  $N_6$ , CCN and RBF in the presence of non-Gaussian noise. For example, for the Autoprice dataset, RMSEs for C2N2,  $N_4$ - $N_6$ , CCN and RBF are 0.2689, 0.2996, 0.2689, 0.3681, 0.2775 and 0.2725, respectively. After adding noise, RMSEs are 0.2770, 0.5521, 0.3610, 0.4295, 0.4768 and 0.9082 respectively. These results confirm that the proposed method is robust to impulsive noise in comparison to the mentioned methods.

5.4.5. Discussion on Table 4

This table compares C2N2 with the networks  $N_4, N_5, N_6$ , CCN and RBF on the Ionosphere, Colon, Leukemia and Dimdata datasets in the presence of impulsive noise. It shows that C2N2 outperforms the other mentioned methods on the Ionosphere, Colon and Leukemia datasets. For the dataset Dimdata, RBF outperforms the proposed method. However, the result is obtained with 1000 nodes for RBF and with an average of 3.5 nodes for C2N2.

5.4.6. Discussion on Tables 5 and 6

These tables compare C2N2 with state-of-the-art constructive networks on the regression and classification datasets. They show that the best results are for C2N2. For the Housing dataset, RMSEs for C2N2, OLSCN and OHLCN are 0.0966, 0.0988 and 0.0993, respectively. In the presence of impulsive noise, RMSEs are 0.0978, 0.2411 and 0.1824. Thus, the proposed method has the best stability among the other mentioned state-of-the-art constructive networks when data samples are contaminated with impulsive noise. From the architecture viewpoint, OLSCN has the most compact architecture. However, it has the

worst training time. Table 6 shows that C2N2 outperforms OLSCN and OHLCN in the presence of impulsive noise in the classification dataset.

#### 5.4.7. Computational Complexity

Let  $L$  be the maximum number of hidden units to be added to the network. For each newly added node, its input parameter is adjusted as specified in Section 4.4.1. The order of adjusting the inverse of the  $d \times d$  matrix is  $d^3$ . Thus,  $h = O(Lk(\min(d^3, N^3) + N^2))$ , where  $k$  is the constant term (number of iterations). Thus, we have

$$h = O\left(L\left(\min\left(d^3, N^3\right) + N^2\right)\right).$$

### 5.5. Comparison

This part compares C2N2 with state-of-the-art robust learning methods on several benchmark datasets. These methods are Robust Least Square SVM (RLS-SVM), MLPMEE and MLPMCC. As mentioned in Section 5.4, and similar to [4], in this part, experiments are performed in 20 trials and average results of RMSE (accuracy for classification) and the number of hidden nodes(#N) are reported in Table 9.

#### 5.5.1. Discussion on Table 9

From the table, we can see that for Pyrim, Prim (noise) and basketball (noise), C2N2 absolutely outperforms the other robust methods in terms of RMSE. For Bodyfat and Bodyfat (noise), C2N2 slightly outperforms other methods in terms of RMSE. Thus, to compare them, we need to check the number of nodes and training times. For both datasets, C2N2 has a fewer number of nodes. In the presence of noise, C2N2 has a better training time in comparison to RLS-LSVM.

Thus, the proposed method outperforms the other methods for these two datasets. Among these six datasets, RLS-SVM only outperforms C2N2 in one dataset, i.e., Basketball; however, it has a worse training time and more nodes. It can be seen that among the robust methods, the proposed method has the most compact architecture.

**Table 9.** Performance comparison of C2N2 and the state-of-the-art robust methods MLPMEE, MLPMCC and RLS-LSVM: benchmark regression dataset.

Dataset	C2N2			MLPMEE			MLPMCC			RLS-LSVM		
	Testing RMSE	#Nodes	Time (s)	Testing RMSE	#Nodes	Time (s)	Testing RMSE	#Nodes	Time (s)	Testing RMSE	#Nodes	Time (s)
Bodyfat	<u>0.00281</u>	<u>5</u>	0.4623	0.0045	10	-	<u>0.00291</u>	40	-	<u>0.00295</u>	101	<u>0.0789</u>
Bodyfat (Noise)	<u>0.00241</u>	<u>4</u>	<u>0.2751</u>	<u>0.00251</u>	10	-	<u>0.00257</u>	10	-	0.00451	101	9.651
Pyrim	<u>0.0488</u>	<u>7</u>	0.2712	0.0798	20	-	0.0882	40	-	0.0817	37.3	<u>0.0352</u>
Pyrim (Noise)	<u>0.05213</u>	<u>4.4</u>	<u>0.1521</u>	<u>0.0586</u>	10	-	0.12034	30	-	0.12345	37	4.495
Basketball	0.12293	<u>5.5</u>	<u>0.2454</u>	0.1352	30	-	0.13114	20	-	<u>0.1143</u>	48	0.3687
Basketball (noise)	<u>0.11981</u>	<u>1.33</u>	<u>0.0238</u>	0.14352	20	-	0.1328	20	-	0.12839	48	22.569

#### 5.5.2. Discussion on Table 10

This table compares the recent work, CCOEN, with the proposed method on three datasets in the presence and absence of noise. According to the table, the proposed method outperforms CCOEN in all cases except the Cloud dataset in the presence of noise where CCOEN has a slightly better performance with more hidden nodes. From the architecture viewpoint, the proposed method has a fewer number of nodes in comparison to CCOEN in most cases. Therefore, correntropy with the Gaussian kernel provides better results in comparison to the sigmoid kernel.

**Table 10.** Performance comparison of C2N2 and the recent work. CCOEN: benchmark regression dataset.

Dataset	C2N2		CCOEN	
	Testing RMSE	#Nodes	Testing RMSE	#Nodes
Abalone	<b>0.075</b>	<b>6</b>	0.090	8.8
Abalone (Noise)	<b>0.079</b>	<b>2.6</b>	<b>0.091</b>	7.8
Cleveland	<b>0.061</b>	<b>4.2</b>	0.791	6.1
Cleveland (Noise)	<b>0.066</b>	<b>2</b>	0.821	8.5
Cloud	<b>0.277</b>	<b>4.2</b>	0.293	4.7
Cloud (noise)	0.302	5.6	<b>0.290</b>	<b>4.8</b>

## 6. Conclusions

In this paper, a new constructive feedforward network is presented that is robust to non-Gaussian noises. Most of the other existing constructive networks are trained based on the mean square error (MSE) objective function and consequently act weak in the presence of non-Gaussian noises, especially impulsive noise. Correntropy is a local similarity measure of two random variables and is successfully used as the objective function for the training of adaptive systems such as adaptive filters. In this paper, this objective function with a Gaussian kernel is utilized to adjust the input and output parameters of the newly added node in a constructive network. It is proved that the new node obtained from the input side optimization is not orthogonal to the residual error of the network. Regarding this fact, correntropy of the residual error converges to its optimum when the error and the activation function are continuous random variables in  $\mathcal{L}^2(\Omega, \mathcal{F}, \mathcal{P})$  space where the triple  $(\Omega, \mathcal{F}, \mathcal{P})$  is considered as a probability space. During the training on datasets, the function form of error is not available; thus, we provide a method to adjust the input and output parameters of the new node from training data samples. The auxiliary variables that appear in input and output side optimization problems decrease the effect of the non-Gaussian noises. For example, for impulsive noise, these variables are close to zero; thus, these data samples have little role in optimizing the parameters of the network. For the MSE-based constructive networks, the data samples that are contaminated by impulsive noises have a great role in optimizing the parameters of the network, and consequently, the network is not robust. The experiments are performed on some synthetic and benchmark datasets. For the synthetic datasets, the experiments are performed in the presence and absence of impulsive noises. We saw that for the datasets that are contaminated by impulsive noises, the proposed method has significantly better performance than the state-of-the-art MSE-based constructive network. For the other synthetic and benchmark datasets, in most cases, the proposed method has satisfactory performance in comparison to the MSE-based constructive network and radial basis function (RBF). Furthermore, C2N2 was compared with state-of-the-art robust learning methods such as MLPME, MLP MCC and the robust version of the Least Square Support Vector Machine and CCOEN. The performances are obtained with compact architectures due to the input parameters being optimized. We also see that correntropy with Gaussian kernel provides better results in comparison to the correntropy with sigmoid kernel.

The use of the correntropy-based function introduced in this research may also benefit networks with other architectures toward enhancing the generalization performance and robustness level. In the context of further research, the validity of similar results can be verified for various classes of neural networks. In addition, since impulsive noise is one of the worst cases of non-Gaussian noise, it can be expected that a different non-Gaussian noise will yield a result between clean data and data with impulsive noise. This should be verified in further experiments.

It is also necessary to point out here other novel modern avenues and similar research directions. For example, ref. [39] delves into modal regression, presenting a statistical

learning perspective that could enrich the discussion on learning algorithms and their efficiency in different noise conditions. In particular, it points out that correntropy-based regression can be interpreted from a modal regression viewpoint when the tuning parameter goes to zero. At the same time, [40] depicts a big picture of correntropy-based regression by showing that with different choices of the tuning parameter, correntropy-based regression learns a location function.

Correntropy not only has inferential properties that could be used for neural network analysis, but another approach could be, for example, cross-sample entropy-based techniques. One such direction was shown to be effective in [41] with reported results of simulation on exchange market datasets.

Finally, it is also worth mentioning that the choice of the algorithm applied for optimizing the objective functions can influence the results. The usage of non-smooth methodology focusing on bundle-based algorithms [42] as a possible efficient tool in machine learning and neural network analysis can also be tested.

**Author Contributions:** All authors contributed to the paper. The experimental part was mainly done by the first author, M.N. Conceptualization was done by M.R. and H.S.Y. Verification and final editing of the manuscript was done by Y.N., A.M. and M.M.M. who played a role of a research director. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** Data and codes can be requested from the first author

**Acknowledgments:** The authors would like to express their thanks to the GA. Hodtani and Ehsan Shams Davodly for their constructive remarks and suggestions.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

C2N2	Correntropy-based constructive neural network
MCC	Maximum correntropy criterion
MSE	Mean square error
MEE	Minimum Error Entropy
EMSE	Excess mean square error
OLS	Orthogonal least square
CCOEN	Cascade correntropy network
MLPMEE	Multi-Layer Perceptron based on Minimum Error Entropy
MLPMCC	Multi-Layer Perceptron based on correntropy
RLS-SVM	Robust Least Square Support Vector Machine
FFN	Feedforward network
RBF	Radial basis function
ITL	Information theoretic learning
CGN	Cascade correlation network
OHLCN	One hidden layer constructive adaptive neural network

## Appendix A

### Appendix A.1. Proof of Lemma 1

**Proof.** Similar to [28], let  $k_L = \frac{\|e_{L-1}\|}{\|g_L\|} \gamma$ ,  $\gamma \in \mathbb{R} - \{0\}$  where

$$\forall g_L \in \mathcal{G} |\cos(\theta_{e_{L-1}, g_L^*})| \geq |\cos(\theta_{e_{L-1}, g_L})|.$$

Thus,

$$\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|e_{L-1} - k_L g_L\|^2}{2\sigma^2}\right) =$$

$$\frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2 + \|e_{L-1}\|^2\gamma^2\frac{\|g_L\|^2}{\|g_L\|^2} - 2\gamma\|e_{L-1}\|\frac{\|g_L\|^2}{\|g_L\|} \cos(\theta_{e_{L-1}, g_L})}{2\sigma^2}\right).$$

Let  $A = \frac{\|g_L\|}{\|g_L\|}$ , we have,

$$\begin{aligned} & \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1} - k_L g_L\|^2}{2\sigma^2}\right) = \\ & \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2(1 + A^2\gamma^2 - 2\gamma A \cos(\theta_{e_{L-1}, g_L}))}{2\sigma^2}\right). \end{aligned}$$

We need to prove that  $k_L \in \mathbb{R} - \{0\}$  exists such that

$$\begin{aligned} & \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2(1 + \gamma^2 - 2\gamma \cos(\theta_{e_{L-1}, g_L^*}))}{2\sigma^2}\right) \geq \\ & \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2(1 + A^2\gamma^2 - 2\gamma A \cos(\theta_{e_{L-1}, g_L}))}{2\sigma^2}\right), \forall g_L \in \mathcal{G}. \end{aligned}$$

Suppose that the inequality above does not hold.

Two possible conditions may happen:

1.  $\cos(\theta_{e_{L-1}, g_L^*}) \geq 0$   
 1.1. If  $\gamma > 0$ , we have

$$\begin{aligned} & \forall \gamma > 0 \quad 2\gamma(\cos(\theta_{e_{L-1}, g_L^*}) - A \cos(\theta_{e_{L-1}, g_L})) \\ & \leq 2\gamma(\cos(\theta_{e_{L-1}, g_L^*}) - A \cos(\theta_{e_{L-1}, g_L})) \leq \gamma^2(1 - A^2), \quad \forall g_L \in \mathcal{G} \\ & \rightarrow \forall \gamma > 0 \quad 2\gamma \cos(\theta_{e_{L-1}, g_L^*})(1 - A) \leq \gamma^2(1 - A^2), \forall g_L \in \mathcal{G}. \end{aligned}$$

Again, there are two possible conditions: First,  $(1 - A) \geq 0$ . Then

$$\begin{aligned} & \rightarrow \forall \gamma > 0 \quad 2 \cos(\theta_{e_{L-1}, g_L^*}) \leq \gamma(1 + A) \quad \forall g_L \in \mathcal{G}. \\ & \rightarrow \forall \gamma > 0 \quad \frac{2 \cos(\theta_{e_{L-1}, g_L^*})}{(1+A)} \leq \gamma, \quad \forall g_L \in \mathcal{G}. \end{aligned}$$

Let  $\gamma = \frac{\cos(\theta_{e_{L-1}, g_L^*})}{(1+A)} \rightarrow \cos(\theta_{e_{L-1}, g_L^*}) \leq 0, \quad \forall g_L \in \mathcal{G}$ . From the assumption and the above inequality, we have  $\cos(\theta_{e_{L-1}, g_L^*}) = 0$ .

This is contradicted by the fact that  $\text{span}(\mathcal{G})$  is dense in  $L^2(\Omega, \mathcal{F}, \mathcal{P})$ . Second:  $(1 - A) \leq 0$ .

$$\begin{aligned} & \rightarrow \forall \gamma > 0 \quad 2 \cos(\theta_{e_{L-1}, g_L^*}) \geq \gamma(1 + A) \quad \forall g_L \in \mathcal{G}. \\ & \rightarrow \forall \gamma > 0 \quad \frac{2 \cos(\theta_{e_{L-1}, g_L^*})}{(1+A)} \geq \gamma \quad \forall g_L \in \mathcal{G}. \end{aligned}$$

$$\text{Let } \gamma = \frac{3 \cos(\theta_{e_{L-1}, g_L^*})}{(1+A)} \rightarrow \cos(\theta_{e_{L-1}, g_L^*}) \leq 0 \quad \forall g_L \in \mathcal{G}.$$

From the assumption above, we have  $\cos(\theta_{e_{L-1}, g_L^*}) = 0$ .

This is contradicted by the fact that  $\text{span}(\mathcal{G})$  is dense in  $L^2(\Omega, \mathcal{F}, \mathcal{P})$ .

2.  $\cos(\theta_{e_{L-1}, g_L^*}) \leq 0$ .

2.1. If  $\gamma < 0$ , we have

$$\begin{aligned} & \forall \gamma < 0 \quad 2\gamma(\cos(\theta_{e_{L-1}, g_L^*}) - A \cos(\theta_{e_{L-1}, g_L})) \leq 2\gamma(\cos(\theta_{e_{L-1}, g_L^*}) - A \cos(\theta_{e_{L-1}, g_L})) \\ & \leq \gamma^2(1 - A^2), \quad \forall g_L \in \mathcal{G}. \end{aligned}$$

$$\rightarrow \forall \gamma < 0 \quad 2\gamma \cos(\theta_{e_{L-1}}, g_L^*)(1 - A) \leq \gamma^2(1 - A^2) \quad \forall g_L \in \mathcal{G}$$

$$\rightarrow \forall \gamma < 0 \quad 2 \cos(\theta_{e_{L-1}}, g_L^*)(1 - A) \leq \gamma(1 - A^2) \forall g_L \in \mathcal{G}.$$

Again, there are two possible conditions: First:  $(1 - A) \geq 0$ . Then

$$\forall \gamma < 0 \quad 2 \cos(\theta_{e_{L-1}}, g_L^*) \leq \gamma(1 + A) \quad \forall g_L \in \mathcal{G},$$

$$\forall \gamma < 0 \quad \frac{2 \cos(\theta_{e_{L-1}}, g_L^*)}{(1+A)} \leq \gamma \forall g_L \in \mathcal{G}.$$

$$\text{Let } \gamma = \frac{3 \cos(\theta_{e_{L-1}}, g_L^*)}{(1+A)} \rightarrow \cos(\theta_{e_{L-1}}, g_L^*) \geq 0, \quad \forall g_L \in \mathcal{G}.$$

$$\in \mathcal{G} \cos(\theta_{e_{L-1}}, g_L^*) = 0.$$

From the assumption, we have  $\cos(\theta_{e_{L-1}}, g_L^*) = 0$ .

This is contradicted by the fact that  $\text{span}(\mathcal{G})$  is dense in  $L^2(\Omega, \mathcal{F}, \mathcal{P})$ . Second:  $(1 - A) \leq 0$

$$\forall \gamma < 0 \quad 2 \cos(\theta_{e_{L-1}}, g_L^*) \geq \gamma(1 + A) \quad \forall g_L \in \mathcal{G}.$$

$$\forall \gamma < 0 \quad \frac{2 \cos(\theta_{e_{L-1}}, g_L^*)}{(1+A)} \geq \gamma \quad \forall g_L \in \mathcal{G}.$$

$$\text{Let } \gamma = \frac{\cos(\theta_{e_{L-1}}, g_L^*)}{(1+A)} \rightarrow \cos(\theta_{e_{L-1}}, g_L^*) \geq 0, \quad \forall g_L \in \mathcal{G}.$$

From the assumption above, we have,  $\cos(\theta_{e_{L-1}}, g_L^*) = 0$ , and this is contradicted by the fact that  $\text{span}(\mathcal{G})$  is dense in  $L^2(\Omega, \mathcal{F}, \mathcal{P})$ . Based on the above arguments, a real number  $\gamma \neq 0$  exists such that the following inequality holds:

$$\begin{aligned} & \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2(1 + \gamma^2 - 2\gamma \cos(\theta_{e_{L-1}}, g_L^*))}{2\sigma^2}\right) \\ & \geq \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2(1 + A^2\gamma^2 - 2\gamma A \cos(\theta_{e_{L-1}}, g_L))}{2\sigma^2}\right), \forall g_L \in \mathcal{G}. \end{aligned}$$

Thus, from Theorem 4, we have

$$\begin{aligned} & E\left(\frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2(1 + \gamma^2 - 2\gamma \cos(\theta_{e_{L-1}}, g_L^*))}{2\sigma^2}\right)\right) \geq \\ & E\left(\frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2(1 + A^2\gamma^2 - 2\gamma A \cos(\theta_{e_{L-1}}, g_L))}{2\sigma^2}\right)\right), \forall g_L \in \mathcal{G}. \end{aligned}$$

Thus, there exists a real number  $k_L \in \mathbb{R} - \{0\}$  such that

$$\begin{aligned} E\left(\frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1} - k_L g_L^*\|^2}{2\sigma^2}\right)\right) & \geq E\left(\frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1} - k_L g_L\|^2}{2\sigma^2}\right)\right), \forall g_L \in \mathcal{G}, \\ & \rightarrow V(g_L^*) \geq V(g_L), \forall g_L \in \mathcal{G}. \end{aligned}$$

This means that  $g_L^{\text{sim}(e)} = g_L^*$  and this completes the proof.  $\square$

#### Appendix A.2. Proof of Theorem 6

**Proof.** Inspired by [3], the proof of this theorem is divided into two parts: First, we prove that the correntropy of the network strictly increases after adding each hidden node, and then we prove that the supremum of the correntropy of the network is  $V_{\max} = \frac{1}{\sqrt{2\pi\sigma}}$ .

Step 1: The correntropies of an SLFN with  $L - 1$  and  $L$  hidden nodes are:

$$V(e_{L-1}) = E\left(\frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2}{2\sigma^2}\right)\right),$$

$$\begin{aligned}
 V(e_L) &= E\left(\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|e_L\|^2}{2\sigma^2}\right)\right) \\
 &= E\left(\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|e_{L-1} - \beta_L^{sim(e)} g_L^{sim(e)}\|^2}{2\sigma^2}\right)\right),
 \end{aligned}$$

respectively.

In the following, it is proved that  $\{k_L, g_L^e\}$  exists such that

$$V(e_L) > V(e_{L-1}).$$

Let

$$k'_L = \eta \|e_{L-1}\|, \eta \in \mathbb{R} - \{0\},$$

then we have

$$V(g_L) = E\left(\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|e_{L-1} - \eta \|e_{L-1}\| g_L\|^2}{2\sigma^2}\right)\right) = E\left(\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{U}{2\sigma^2}\right)\right),$$

where

$$U = \|e_{L-1}\|^2 + \eta^2 \|e_{L-1}\|^2 \|g_L\|^2 - 2\eta \|e_{L-1}\|^2 \|g_L\| \cos(\theta_{e_{L-1}, g_L})$$

Thus,

$$V(g_L) = E\left(\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|e_{L-1}\|^2 \left( (1 + \eta^2 \|g_L\|^2) - 2\eta \|g_L\| \cos(\theta_{e_{L-1}, g_L}) \right)}{2\sigma^2}\right)\right).$$

We need to prove that  $g_L \in \mathcal{G}$  and  $\eta \in \mathbb{R} - \{0\}$  exist such that

$$V(g_L) > V(e_{L-1}).$$

Suppose that there are no  $\eta \in \mathbb{R} - \{0\}$  and  $g_L \in \mathcal{G}$  such that

$$\begin{aligned}
 \exp\left(-\frac{\|e_{L-1}\|^2 \left( (1 + \eta^2 \|g_L\|^2) - 2\eta \|g_L\| \cos(\theta_{e_{L-1}, g_L}) \right)}{2\sigma^2}\right) &\leq \exp\left(-\frac{\|e_{L-1}\|^2}{2\sigma^2}\right), \\
 \rightarrow \exp\left(-\frac{\|e_{L-1}\|^2}{2\sigma^2}\right) \exp\left(\frac{\|e_{L-1}\|^2 \left( (1 + \eta^2 \|g_L\|^2) - 2\eta \|g_L\| \cos(\theta_{e_{L-1}, g_L}) \right)}{2\sigma^2}\right) &\geq 1,
 \end{aligned}$$

$\forall g_L \in \mathcal{G}$  and  $\forall \eta \in \mathbb{R} - \{0\}$ .

Then the following inequality holds,

$$\exp\left(\frac{\|e_{L-1}\|^2 \left( (\eta^2 \|g_L\|^2) - 2\eta \|g_L\| \cos(\theta_{e_{L-1}, g_L}) \right)}{2\sigma^2}\right) \geq \exp(0),$$

$\forall g_L \in \mathcal{G} \quad \forall \eta \in \mathbb{R} - \{0\}$ .

$$\rightarrow \|e_{L-1}\|^2 \left( (\eta^2 \|g_L\|^2) - 2\eta \|g_L\| \cos(\theta_{e_{L-1}, g_L}) \right) \geq 0,$$

$\forall g_L \in \mathcal{G}$  and  $\forall \eta \in \mathbb{R} - \{0\}$

$$\rightarrow \eta^2 \|g_L\|^2 - 2\eta \|g_L\| \cos(\theta_{e_{L-1}, g_L}) \geq 0,$$

$\forall g_L \in \mathcal{G}$  and  $\forall \eta \in \mathbb{R} - \{0\}$ .

Let

$$\eta = \frac{\cos(\theta_{e_{L-1}, g_L})}{\|g_L\|}, \quad \cos^2(\theta_{e_{L-1}, g_L}) - 2\cos^2(\theta_{e_{L-1}, g_L}) \geq 0 \quad \forall g_L \in \mathcal{G},$$

$$\rightarrow \cos^2(\theta_{e_{L-1}, g_L}) \leq 0 \quad \forall g_L \in \mathcal{G}.$$

Thus,  $\cos^2(\theta_{e_{L-1}, g_L}) = 0 \quad \forall g_L \in \mathcal{G}$ . This is contradictory to  $\text{span}(\mathcal{G})$  being dense in  $L^2(\Omega, \mathcal{F}, \mathcal{P})$ ; thus, we have

$$\exists g_L \in \mathcal{G}, \exists k'_L \in \mathbb{R} - \{0\} \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1} - k'_L g_L\|^2}{2\sigma^2}\right) > \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2}{2\sigma^2}\right).$$

Based on Theorem 4, the following inequality holds with probability one:

$$\exists g_L \in \mathcal{G}, \exists k'_L \in \mathbb{R} - \{0\}$$

$$E\left(\frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1} - k'_L g_L\|^2}{2\sigma^2}\right)\right) > \left(\frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\|e_{L-1}\|^2}{2\sigma^2}\right)\right), \text{ i.e.,}$$

$$\exists g_L \in \mathcal{G}, \exists k'_L \in \mathbb{R} - \{0\}, \quad E(k(e_{L-1} - k'_L g_L)) > E(k(e_{L-1}))$$

almost surely.

Based on the above argument, with probability one, we have:  $V(e_L) > V(e_{L-1})$

Based on Theorem 5, since the correntropy is strictly increasing, with probability one it converges to its supremum.

Step 2: We know that

$$V_L^\beta = E\left(\exp\left(-\frac{\|e_{L-1} - \beta_L g_L^{\text{sim}(e)}\|^2}{2\sigma^2}\right)\right) = E\left(\sup_{\alpha \in \mathcal{R}^-} \left(\alpha \frac{\|e_{L-1} - \beta_L g_L^{\text{sim}(e)}\|^2}{2\sigma^2} - \phi(\alpha)\right)\right),$$

and according to Proposition 1, we have

$$\alpha = -G\left(\frac{\|e_{L-1} - \beta_L g_L^{\text{sim}(e)}\|^2}{2\sigma^2}\right),$$

There is

$$V_{Lmax}^\beta = E\left(\left(\alpha \frac{\|e_{L-1} - \beta_L g_L^{\text{sim}(e)}\|^2}{2\sigma^2} - \phi(\alpha)\right)\right).$$

Therefore, the optimum  $\beta_L$  is

$$\beta_L = \frac{E(\gamma e_{L-1} g_L)}{E(\gamma g_L^2)}.$$

In the previous step, we showed that correntropy converges; the norm of error converges and  $\gamma$  converges to a constant term. In the case of constant  $\gamma$ , similar to [3], the error sequence constitutes a Cauchy sequence and because the mentioned probability space is complete,  $\exists e^* \in \mathcal{L}^p(\Omega, \mathcal{F}, \mathcal{P})$ . Therefore,  $\forall L > N, \forall g \in \mathcal{G}$

$$e_L \rightarrow e^*,$$

$$\lim_{L \rightarrow \infty} \frac{E(e_{L-1}g_L)^2}{E(g_L^2)} = 0.$$

Thus, similar to [3], we have

As we know,  $E(e^*g) = 0, \forall g \in \mathcal{G}$ , and we have

$$\lim_{L \rightarrow \infty} E(e_{L-1}g) = 0, \forall g \in \mathcal{G}$$

and

$$\|e^*\| = 0.$$

Based on Theorem 3 and  $\lim_{L \rightarrow \infty} \|e_L\| = 0$ , we have  $\lim_{L \rightarrow \infty} E(k(e_L)) = E(\lim_{L \rightarrow \infty} (k(e_L))) = E\left(\frac{1}{\sqrt{2\pi\sigma}} e^{-\lim_{L \rightarrow \infty} \|e_L\|^2}\right) = E(k(0)) = \frac{1}{\sqrt{2\pi\sigma}} = V_{max}$ . Based on step 1 and step 2, we have  $\lim_{L \rightarrow \infty} V(e_L) = V_{max}$  almost surely.

This completes the proof.  $\square$

## References

1. Erdogmus, D.; Principe, J.C. An error-entropy minimization algorithm for supervised training of nonlinear adaptive systems. *Signal Process. IEEE Trans.* **2002**, *50*, 1780–1786. [\[CrossRef\]](#)
2. Fahlman, S.E.; Lebiere, C. The cascade-correlation learning architecture. In Proceedings of the Advances in Neural Information Processing Systems 2, NIPS Conference, Denver, CO, USA, 27–30 November 1989; pp. 524–532.
3. Kwok, T.-Y.; Yeung, D.-Y. Objective functions for training new hidden units in constructive neural networks. *Neural Netw. IEEE Trans.* **1997**, *8*, 1131–1148. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Huang, G.; Song, S.; Wu, C. Orthogonal least squares algorithm for training cascade neural networks. *Circuits Syst. Regul. Pap. IEEE Trans.* **2012**, *59*, 2629–2637. [\[CrossRef\]](#)
5. Ma, L.; Khorasani, K. New training strategies for constructive neural networks with application to regression problems. *Neural Netw.* **2004**, *17*, 589–609. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Ma, L.; Khorasani, K. Constructive feedforward neural networks using Hermite polynomial activation functions. *Neural Netw. IEEE Trans.* **2005**, *16*, 821–833. [\[CrossRef\]](#) [\[PubMed\]](#)
7. Reed, R. Pruning algorithms—a survey. *Neural Netw. IEEE Trans.* **1993**, *4*, 740–747. [\[CrossRef\]](#) [\[PubMed\]](#)
8. Castellano, G.; Fanelli, A.M.; Pelillo, M. An iterative pruning algorithm for feedforward neural networks. *Neural Netw. IEEE Trans.* **1997**, *8*, 519–531. [\[CrossRef\]](#) [\[PubMed\]](#)
9. Engelbrecht, A.P. A new pruning heuristic based on variance analysis of sensitivity information. *Neural Netw. IEEE Trans.* **2001**, *12*, 1386–1399. [\[CrossRef\]](#)
10. Zeng, X.; Yeung, D.S. Hidden neuron pruning of multilayer perceptrons using a quantified sensitivity measure. *Neurocomputing* **2006**, *69*, 825–837. [\[CrossRef\]](#)
11. Sakar, A.; Mammone, R.J. Growing and pruning neural tree networks. *Comput. IEEE Trans.* **1993**, *42*, 291–299. [\[CrossRef\]](#)
12. Huang, G.-B.; Saratchandran, P.; Sundararajan, N. A generalized growing and pruning RBF (GGAPRBF) neural network for function approximation. *Neural Netw. IEEE Trans.* **2005**, *16*, 57–67. [\[CrossRef\]](#)
13. Huang, G.-B.; Saratchandran, P.; Sundararajan, N. An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks. *Syst. Man. Cybern. Part Cybern. IEEE Trans.* **2004**, *34*, 2284–2292. [\[CrossRef\]](#) [\[PubMed\]](#)
14. Wu, X.; Rozycki, P.; Wilamowski, B.M. A Hybrid Constructive Algorithm for Single-Layer Feedforward Networks Learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *26*, 1659–1668. [\[CrossRef\]](#) [\[PubMed\]](#)
15. Santamaría, I.; Pokharel, P.P.; Principe, J.C. Generalized correlation function: Definition, properties, and application to blind equalization. *Signal Process. IEEE Trans.* **2006**, *54*, 2187–2197. [\[CrossRef\]](#)
16. Liu, W.; Pokharel, P.P.; Principe, J.C. Correntropy: Properties and applications in non-Gaussian signal processing. *Signal Process. IEEE Trans.* **2007**, *55*, 5286–5298. [\[CrossRef\]](#)
17. Bessa, R.J.; Miranda, V.; Gama, J. Entropy and correntropy against minimum square error in offline and online three-day ahead wind power forecasting. *Power Syst. IEEE Trans.* **2009**, *24*, 1657–1666. [\[CrossRef\]](#)
18. Singh, A.; Principe, J.C. Using correntropy as a cost function in linear adaptive filters. In Proceedings of the 2009 International Joint Conference on Neural Networks, Atlanta, GA, USA, 14–19 June 2009; pp. 2950–2955.
19. Shi, L.; Lin, Y. Convex Combination of Adaptive Filters under the Maximum Correntropy Criterion in Impulsive Interference. *Signal Process. Lett. IEEE* **2014**, *21*, 1385–1388. [\[CrossRef\]](#)
20. Zhao, S.; Chen, B.; Principe, J.C. Kernel adaptive filtering with maximum correntropy criterion. In Proceedings of the 2011 International Joint Conference on Neural Networks, San Jose, CA, USA, 31 July–5 August 2011; pp. 2012–2017.

21. Wu, Z.; Peng, S.; Chen, B.; Zhao, H. Robust Hammerstein Adaptive Filtering under Maximum Correntropy Criterion. *Entropy* **2015**, *17*, 7149–7166. [[CrossRef](#)]
22. Chen, B.; Wang, J.; Zhao, H.; Zheng, N.; Principe, J.C. Convergence of a fixed-point algorithm under Maximum Correntropy Criterion. *Signal Process. Lett. IEEE* **2015**, *22*, 1723–1727. [[CrossRef](#)]
23. Chen, B.; Xing, L.; Liang, J.; Zheng, N.; Principe, J.C. Steady-state mean-square error analysis for adaptive filtering under the maximum correntropy criterion. *Signal Process. Lett. IEEE* **2014**, *21*, 880–884.
24. Chen, L.; Qu, H.; Zhao, J.; Chen, B.; Principe, J.C. Efficient and robust deep learning with Correntropy-induced loss function. *Neural Comput. Appl.* **2015**, *27*, 1019–1031. [[CrossRef](#)]
25. Singh, A.; Principe, J.C. A loss function for classification based on a robust similarity metric. In Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, Spain, 18–23 July 2010; pp. 1–6.
26. Feng, Y.; Huang, X.; Shi, L.; Yang, Y.; Suykens, J.A. Learning with the maximum correntropy criterion induced losses for regression. *J. Mach. Learn. Res.* **2015**, *16*, 993–1034.
27. Chen, B.; Principe, J.C. Maximum correntropy estimation is a smoothed MAP estimation. *Signal Process. Lett. IEEE* **2012**, *19*, 491–494. [[CrossRef](#)]
28. Nayyeri, M.; Yazdi, H.S.; Maskooki, A.; Rouhani, M. Universal Approximation by Using the Correntropy Objective Function. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 4515–4521. [[CrossRef](#)]
29. Athreya, K.B.; Lahiri, S.N. *Measure Theory and Probability Theory*; Springer Science & Business Media: New York, NY, USA, 2006.
30. Fournier, N.; Guillin, A. On the rate of convergence in Wasserstein distance of the empirical measure. *Probab. Theory Relat. Fields* **2015**, *162*, 707–738. [[CrossRef](#)]
31. Leshno, M.; Lin, V.Y.; Pinkus, A.; Schocken, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Netw.* **1993**, *6*, 861–867. [[CrossRef](#)]
32. Yuan, X.-T.; Hu, B.-G. Robust feature extraction via information theoretic learning. In Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, QC, Canada, 14–18 June 2009; pp. 1193–1200.
33. Klenke, A. *Probability Theory: A Comprehensive Course*; Springer Science & Business Media: New York, NY, USA, 2013.
34. Rudin, W. *Principles of Mathematical Analysis*; McGraw-Hill: New York, NY, USA, 1964; Volume 3.
35. Yang, X.; Tan, L.; He, L. A robust least squares support vector machine for regression and classification with noise. *Neurocomputing* **2014**, *140*, 41–52. [[CrossRef](#)]
36. Newman, D.; Hettich, S.; Blake, C.; Merz, C.; Aha, D. *UCI Repository of Machine Learning Databases*; Department of Information and Computer Science, University of California: Irvine, CA, USA, 1998. Available online: <https://archive.ics.uci.edu/> (accessed on 29 November 2023).
37. Meyer, M.; Vlachos, P. Statlib. 1989. Available online: <https://lib.stat.cmu.edu/datasets/> (accessed on 29 November 2023).
38. Pokharel, P.P.; Liu, W.; Principe, J.C. A low complexity robust detector in impulsive noise. *Signal Process.* **2009**, *89*, 1902–1909. [[CrossRef](#)]
39. Feng, Y.; Fan, J.; Suykens, J.A. A Statistical Learning Approach to Modal Regression. *J. Mach. Learn. Res.* **2020**, *21*, 1–35.
40. Feng, Y. New Insights into Learning with Correntropy-Based Regression. *Neural Comput.* **2021**, *33*, 157–173. [[CrossRef](#)]
41. Ramirez-Parietti, I.; Contreras-Reyes, J.E.; Idrovo-Aguirre, B.J. Cross-sample entropy estimation for time series analysis: A nonparametric approach. *Nonlinear Dyn.* **2021**, *105*, 2485–2508. [[CrossRef](#)]
42. Bagirov, A.; Karmitsa, N.; Mäkelä, M.M. *Introduction to Nonsmooth Optimization: Theory, Practice and Software*; Springer International Publishing: Cham, Switzerland; Heidelberg, Germany, 2014; Volume 12.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.