

The convergence of a generalized convex neural network

Lei Chen^a, Yilin Wang^b, Yunhe Wu^c, Lixiao Zhang^a

^aKitchen & Water Heater Appliance Division, Midea Group, China; ^bCollege of Aeronautics and Astronautics, Shenyang Aerospace University, China; ^cCollege of Information Science and Engineering, Northeastern University, China

ARTICLE HISTORY

Compiled November 7, 2023

ABSTRACT

The convergence of neural networks is a central area of research that is essential in understanding the universal approximation capability and the structural complexity of these systems. In this study, we investigate a generalized convex incremental iteration method. This iteration method is presented in a more general form than previous studies, and is capable of scraping a broader range of weight parameters. Additionally, we systematically demonstrate the convergence rate of the convex iteration. Our findings are also easily extensible to deep convolutional neural networks, which helps to explain their effectiveness in various real-world applications. Furthermore, we provide a discrete statistical perspective to address issues of input data non-compactness and unknowability of the objective function in practical settings. To support our conclusions, we propose two algorithms, namely back propagation and random search, the latter of which can prevent the network from getting stuck in a local minimum during training. Finally, we present results from several regression problems, which indicate that our algorithms exhibit superior performance and are consistent with our theoretical predictions. These results provide a more comprehensive understanding of the convergence of neural networks and its significance in the universal approximation capability of these systems.

KEYWORDS

Generalized convex iteration; convergence rate; universal approximation; neural networks

1. Introduction

Deep neural networks (DNNs) have become a popular topic of research due to their exceptional performance in a wide range of tasks [1–3]. Despite their success, the complex and deep structures of DNNs often elude clear mathematical explanations. A commonly cited reason for the outstanding performance of DNNs is their universal approximation capability, which enables them to approximate target functions with high accuracy [4–18]. This capability has been extensively studied in the literature and it has been shown that a well-designed neural network with an increased number of hidden neurons can lead to an approximation of the target function. However, researchers are also concerned with the convergence rate of DNNs. The convergence rate is important as it allows for the estimation of the size of the network, and for determining the optimal balance between computation burden and target accuracy. Furthermore, the convergence rate can serve as an indirect verification of the universal approximation capability of DNNs. Many theories have been proposed to study the convergence rate of DNNs, with many of them based on the classical convex structure, as

they possess desirable mathematical properties in function space [12,19–21].

This paper presents a comprehensive examination of a generalized convex incremental neural network. Building on the work presented in [21], we provide a thorough proof of the convergence rate for a more general form of the convex iteration than the one previously discussed in [21]. Our theory offers a wider range of options for the convex iteration, and several previous convex design structures are shown to be special cases of our theory. Additionally, networks that utilize our theory exhibit the same convergence rate. From the perspective of universal approximation, our theory reinforces the universal approximation capability of single hidden layer neural networks. Although single hidden layer neural network structures are not currently prevalent in the field, our conclusions can be easily extended to deep neural networks, including convolutional neural networks. It is known that deep convolution has a larger receptive field than shallow convolution, and as the number of layers in a convolutional neural network increases, the deep convolutions can cover the entire input range. In other words, these deep convolutions function like single-layer fully connected networks, making our research relevant to deep neural networks as well. Furthermore, our convergence rate theory also helps explain why single hidden layer neural networks are able to perform universal approximation.

In order to validate our theories, it is imperative to provide supporting algorithms that can be implemented in practice. Based on our theoretical descriptions, it is necessary to calculate the minimal residual error of target functions at each step. However, due to the incompleteness of input data and the unpredictability of objective functions in real-world applications, it is infeasible to obtain an exact mathematical expression of the minimal residual error. As a result, we also provide discrete form descriptions of our theories, utilizing limiting inputs to approximate the minimal residual error. To showcase the effectiveness of our generalized convex iteration algorithm, we implement it using two methods: back propagation (BP) and random searching. The random searching method does not require the calculation of gradient of activation functions, which enables it to circumvent the limitations of local minima, vanishing gradients, and exploding gradients associated with BP. Despite the fact that the input data may not encompass the entire input space, our generalized algorithms still exhibit good performance in various regression problems, and the overall testing errors decrease as the number of hidden neurons increases. All simulation results are consistent with our theories.

This paper is structured as follows. In Section 2, we provide an overview of fundamental concepts related to function space, including a convex incremental iteration and a classical convex convergence lemma which serve as the foundation for the subsequent sections. In Section 3, we present the generalized convex iteration and provide a detailed proof of its convergence rate. To address the practical challenges of incomplete input data and unknown objective functions, we present corresponding discrete approximation theories in Section 4. In order to validate our proposed theories, we implement them using backpropagation and random neuron searching methods in Section 5. The performance of our method is evaluated using a variety of classical regression problems in Section 6. We conclude with a summary of the main findings and suggestions for future research in Section 7.

2. Preliminary

A single hidden layer neural network with n neurons is denoted as $f_n(\mathbf{x}) = \sum_{i=1}^n \alpha_i g(\mathbf{w}_i, b_i, \mathbf{x})$, where \mathbf{w}_i is the hidden neuron weight, b_i is the hidden neuron bias, α_i is the weight connecting the i -th hidden neuron to the output neuron, g is the activation function and $g(\mathbf{w}_i, b_i, \mathbf{x})$ is the output of the i -th hidden neuron for the input \mathbf{x} .

We say that a function f is integrable, i.e., $\|f\|^2 = \int_X \|f(\mathbf{x})\|^2 d\mathbf{x} < \infty$, where X is

a compact subset in the d -dimensional Euclidean space \mathbf{R}^d . Let f_n denote an incremental convex feedforward neural network with n hidden neurons, i.e.,

$$f_n = \alpha_n f_{n-1} + \bar{\alpha}_n g_n \quad (1)$$

where $0 \leq \alpha_n \leq 1$, $\bar{\alpha}_n = 1 - \alpha_n$ and $\alpha_1 = 1$. The below lemma demonstrates the convergence rate of a specified convex iteration (1):

Lemma 2.1. [21] *Let G be a subset of a Hilbert space H , with $\|g\| \leq B$ for each $g \in G$. Let $m = \inf_{F \in \text{co}(G)} \|F - f\|$, where $f \in H$ and $\text{co}(G)$ is the convex hull of G .*

Then for every $c > \sqrt{B^2 + m^2}$,

$$\|f_n - f\| \leq m + \frac{c}{\sqrt{n}}$$

where f_n , $n = 1, 2, \dots$, is any sequence chosen to satisfy

$$\|f_1 - f\|^2 \leq \inf_{g \in G} \|g - f\|^2 + \varepsilon_1$$

and, for $n \geq 2$,

$$\|f_n - f\|^2 \leq \inf_{g \in G} \|\alpha_n f_{n-1} + \bar{\alpha}_n g - f\|^2 + \varepsilon_n$$

where $\alpha_n = \frac{n-1}{n}$, $\bar{\alpha}_n = \frac{1}{n}$ and $\varepsilon_n \leq \frac{c^2 - B^2 - m^2}{n^2}$.

Lemma 2.1 provides a theoretical framework for determining the convergence rate of a specified convex neural network. Motivated by Lemma 2.1, we propose to expand the range of parameter selection for the convex iteration, while maintaining the same level of convergence rate. Additionally, it should be noted that the calculation of an infimum is required for each iteration. However, in real-world applications, the objective function is often unknown, and the exact integral expression is difficult to calculate due to the incompleteness of input data. To address this challenge, this paper presents an alternative implementation solution.

Definition 2.2. The hidden neuron $\{g_n = g(\mathbf{w}_n, b_n, \mathbf{x})\}$ is said randomly generated if the corresponding parameters (\mathbf{w}_n, b_n) are randomly generated following the uniform distribution probability.

The above definition is utilized to generate a set of hidden neuron weights for the random searching algorithm.

3. Generalized Convex Networks

In this section, we will prove a kind of generalized incremental convex iteration with the same convergence rate as that of Lemma 2.1.

Theorem 3.1. *Let G be a subset of a Hilbert space H , with $\|g\| \leq B$ for each $g \in G$. Let $m = \inf_{F \in \text{co}(G)} \|F - f\|$, where $f \in H$ and $\text{co}(G)$ is the convex hull of G . For every*

$c > (p+1)\sqrt{B^2 + m^2}$ and $p > -1$, there exists a sequence $f_n = \alpha_n f_{n-1} + \bar{\alpha}_n g_n$ such that

$$\|f_n - f\| \leq m + \frac{c}{\sqrt{n}}$$

where $\alpha_n = \frac{n-1}{n+p}$ and $\bar{\alpha}_n = \frac{p+1}{n+p}$.

Proof. Similar to the proof of Lemma 2.1, for any $\alpha \in [0, 1]$, we have

$$\inf_{g \in \mathcal{G}} \|\alpha_n F + \bar{\alpha} g - f\|^2 \leq \alpha^2 \|F - f\|^2 + 2\alpha\bar{\alpha}m\|F - f\| + \bar{\alpha}^2(B^2 + m^2) \quad (2)$$

where $\bar{\alpha} = 1 - \alpha$.

Set $\alpha_n = \frac{n-1}{n+p}$ and $\bar{\alpha}_n = \frac{p+1}{n+p}$. When $n = 1$, we obtain $\alpha_1 = 0$ and $\bar{\alpha}_1 = 1$, so equation (2) naturally holds.

When $n > 1$, for any $q > -1$, we have $\alpha_n \in (0, 1)$ and $\bar{\alpha}_n \in (0, 1)$. Assume that the desired inequality holds for f_{n-1} , i.e.,

$$\|f_{n-1} - f\|^2 \leq \left(m + \frac{c}{\sqrt{n-1}}\right)^2 \quad (3)$$

Based on Lemma 2.1 and (2), for the n -th iteration, we have

$$\begin{aligned} \|f_n - f\|^2 &\leq \inf_{g \in \mathcal{G}} \|\alpha_n f_{n-1} + \bar{\alpha}_n g - f\|^2 + \varepsilon_n \\ &\leq \alpha_n^2 \|f_{n-1} - f\|^2 + 2\alpha_n \bar{\alpha}_n m \|f_{n-1} - f\| + \bar{\alpha}_n^2 (B^2 + m^2) + \varepsilon_n \\ &= \frac{(n-1)^2}{(n+p)^2} \|f_{n-1} - f\|^2 + 2 \frac{(n-1)(p+1)}{(n+p)^2} m \|f_{n-1} - f\| \\ &\quad + \frac{(p+1)^2}{(n+p)^2} (B^2 + m^2) + \varepsilon_n \end{aligned}$$

With equation (3), the above formula can be modified as

$$\begin{aligned} \|f_n - f\|^2 &\leq \frac{(n-1)^2}{(n+p)^2} \left(m + \frac{c}{\sqrt{n-1}}\right)^2 + 2 \frac{(n-1)(p+1)}{(n+p)^2} m \left(m + \frac{c}{\sqrt{n-1}}\right) \\ &\quad + \frac{(p+1)^2}{(n+p)^2} (B^2 + m^2) + \varepsilon_n \\ &= m^2 \left[\frac{(n-1)^2 + 2(n-1)(p+1)}{(n+p)^2} \right] + 2mc \left[\frac{(n-1)^2 + (n-1)(p+1)}{(n+p)^2 \sqrt{n-1}} \right] \\ &\quad + \frac{n-1}{(n+p)^2} c^2 + \frac{(p+1)^2}{(n+p)^2} (B^2 + m^2) + \varepsilon_n \\ &\leq m^2 \frac{(n-1+p+1)^2}{(n+p)^2} + 2mc \left[\frac{(n-1)^2 + (n-1)(p+1)}{(n+p)^2 \sqrt{n-1}} \right] \\ &\quad + \frac{n-1}{(n+p)^2} c^2 + \frac{(p+1)^2}{(n+p)^2} (B^2 + m^2) + \varepsilon_n \\ &= m^2 + 2mc \frac{\sqrt{n-1}}{n+p} + \frac{n-1}{(n+p)^2} c^2 + \frac{(p+1)^2}{(n+p)^2} (B^2 + m^2) + \varepsilon_n \quad (4) \end{aligned}$$

For the second element in the right equation (4), we set $h_1(p) = \frac{\sqrt{n-1}}{n+p}$, thus $h_1(p)$ is mono-decreasing function for p . When $p = 0$, $h_1(0) = \frac{\sqrt{n-1}}{n} < \frac{1}{\sqrt{n}}$, so for any $p \geq 0$, $h_1(p) < \frac{1}{\sqrt{n}}$, then when $p \geq 0$, equation (4) can be modified as

$$\|f_n - f\|^2 \leq m^2 + \frac{2c}{\sqrt{n}}m + \frac{n-1}{(n+p)^2}c^2 + \frac{(p+1)^2}{(n+p)^2}(B^2 + m^2) + \varepsilon_n \quad (5)$$

So for any $p \geq 0$ and the third element in the right equation (5), we set $h_2(p) = \frac{n-1}{(n+p)^2}c^2 + \frac{(p+1)^2}{(n+p)^2}(B^2 + m^2)$, so we have

$$\begin{aligned} h_2(p) &= \frac{2(p+1)(B^2 + m^2)(n+p)^2}{(n+p)^4} - \frac{2[(n-1)c^2 + (p+1)^2(B^2 + m^2)]}{(n+p)^4} \\ &= 2 \frac{(n-1)[(p+1)(B^2 + m^2) - c^2]}{(n+p)^3} \end{aligned}$$

When $c^2 > (p+1)(B^2 + m^2)$, $h_2(p) < 0$, thus $h_2(p)$ is a mono-decreasing function. $h_2(0) = \frac{(n-1)c^2 + B^2 + m^2}{n^2}$, so when $p \geq 0$, equation (5) can be modified as

$$\|f_n - f\|^2 \leq m^2 + \frac{2c}{\sqrt{n}}m + \frac{(n-1)c^2 + B^2 + m^2}{n^2} + \varepsilon_n \quad (6)$$

Since $\varepsilon_n \leq \frac{c^2 - B^2 - m^2}{n^2}$, equation (6) can be modified as

$$\begin{aligned} \|f_n - f\|^2 &\leq m^2 + \frac{2c}{\sqrt{n}}m + \frac{(n-1)c^2 + B^2 + m^2}{n^2} + \frac{c^2 - B^2 - m^2}{n^2} \\ &= m^2 + \frac{2c}{\sqrt{n}}m + \frac{c^2}{n} \\ &= \left(m + \frac{c}{\sqrt{n}}\right)^2 \end{aligned}$$

So we have

$$\|f_n - f\| \leq m + \frac{c}{\sqrt{n}}$$

The theorem is proved. \square

Remark 1: The convex iteration $\frac{n-1}{n+1}f_{n-1} + \frac{2}{n+1}g$ in Lemma 2.1 can be regarded as a special case for the generalized convex iteration in Theorem 3.1, where we add a parameter p to enlarge the whole iteration parameter range. In [22], the similar generalized convex convergence rate is analyzed, but our convex structure is wider than its. Our Theorem 3.1 takes a novel proof, which enlarges the scope of convex iteration $p \geq -1$, which allows a wider freedom selection of p . Moreover, we present a new convergence rate without the square expression so that the whole expression is simpler.

If $co(G)$ is dense in Hilbert space H or target functions located in $co(G)$, we naturally obtain $m = 0$, i.e., $\lim_{n \rightarrow \infty} \|f_n - f\| = 0$. So we naturally obtain the following corollary:

Corollary 3.2. *If $\text{co}(G)$ is dense in Hilbert space H or target functions located in $\text{co}(G)$, i.e., $f \in \text{co}(G)$, then for every $c > (p+1)B$ and $p > -1$, there exists a sequence $f_n = \alpha_n f_{n-1} + \bar{\alpha}_n g_n$ such that*

$$\|f_n - f\| \leq \frac{c}{\sqrt{n}}$$

where $\alpha_n = \frac{n-1}{n+p}$, $\bar{\alpha}_n = \frac{p+1}{n+p}$ and $\|g\| \leq B$.

Remark 2: Corollary 3.2 illustrates the universal approximation capability of neural networks from an alternative perspective. The similar density condition can be found in previous studies such as [17,18,20]. Our conclusion is consistent with previous studies, providing further evidence for the universality of neural networks.

Remark 3: Theorem 3.1 and Corollary 3.2 can be easily extended to deep convolutional neural networks, as a deep small convolution is equivalent to a shallow big convolution. For example, a two-layer 3x3 convolution is equivalent to a one-layer 5x5 convolution. In a traditional single hidden layer neural network, the input is a d -dimensional vector. However, by using enough deep layers of 3x1 convolutions to construct the network, the entire input range can be covered, effectively functioning as a single hidden layer neural network. Thus, our theories provide a clear explanation for the universal approximation capability of deep convolutional neural networks.

4. Discrete Approximation

In the previous section, we provided a theoretical proof for the convergence rate of neural networks in Theorem 3.1. However, in practical applications, obtaining precise representations is difficult due to the incompleteness of input data and the unknown nature of objective functions. To address this, we provide an indirect demonstration of the convergence rate of neural networks. Our theorem provides a mathematical expression for convergence, however, obtaining the infimum $\inf_{F \in \text{co}(G)} \|F - f\|$ is not always possible. To validate our conclusion, we use random searching methods to approximate the infimum. In this section, we first introduce definitions related to the inner product representation of discrete data and the random neuron description, and then provide a discrete convergence description.

Definition 4.1. [23] We assume that K inputs follow a uniform probability distribution, and for $u, v \in L^2(X)$, the inner product $\langle u, v \rangle$ is defined by $\langle u, v \rangle = \int_X u(\mathbf{x})v(\mathbf{x})d\mathbf{x}$. Its discrete representation is defined as

$$\langle u(\mathbf{x}), v(\mathbf{x}) \rangle = \lim_{K \rightarrow \infty} \sum_{i=1}^K u(\mathbf{x}_i)v(\mathbf{x}_i)$$

Subsequently, we utilize the random searching method to approximate the minimum residual error and construct the entire iteration process.

Theorem 4.2. *Assume the inputs $\{(\mathbf{x}_i, i = 1, 2, \dots, K)\}$ generated with a uniform probability distribution, where K is the number of training samples. Let G be a subset of a Hilbert space H , with $\|g\| \leq B$ for each $g \in G$. Let $m = \inf_{F \in \text{co}(G)} \|F - f\|$, where $f \in H$ and $\text{co}(G)$ is the convex hull of G . For every $c > (p+1)\sqrt{B^2 + m^2}$ and $p > -1$, there exists a*

sequence $f_n = \alpha_n f_{n-1} + \bar{\alpha}_n g_n$ such that

$$\lim_{n, K \rightarrow \infty} \left| \sum_{i=1}^K (f_n(\mathbf{x}_i) - f(\mathbf{x}_i))^2 - m^2 \right| = 0 \quad (7)$$

where $\alpha_n = \frac{n-1}{n+p}$ and $\bar{\alpha}_n = \frac{p+1}{n+p}$.

Proof. Formula (7) can be modified as

$$\begin{aligned} & \lim_{n, K \rightarrow \infty} \left| \sum_{i=1}^K (f_n(\mathbf{x}_i) - f(\mathbf{x}_i))^2 - m^2 \right| \\ & \leq \lim_{n, K \rightarrow \infty} \left| \sum_{i=1}^K (f_n(\mathbf{x}_i) - f(\mathbf{x}_i))^2 - \|f_n - f\|^2 \right| + \lim_{n \rightarrow \infty} |\|f_n - f\|^2 - m^2| \end{aligned}$$

Based on the above definitions, we have

$$\lim_{K \rightarrow \infty} \left| \sum_{i=1}^K (f_n(\mathbf{x}_i) - f(\mathbf{x}_i))^2 - \|f_n - f\|^2 \right| = 0 \quad (8)$$

Based on Theorem 3.1, we know

$$\lim_{n \rightarrow \infty} \|f_n - f\| - m = 0$$

i.e.,

$$\lim_{n \rightarrow \infty} \|f_n - f\|^2 - m^2 = 0 \quad (9)$$

Combining formula (9) and formula (8), we naturally obtain

$$\lim_{n, K \rightarrow \infty} \left| \sum_{i=1}^K (f_n(\mathbf{x}_i) - f(\mathbf{x}_i))^2 - m^2 \right| = 0$$

The theorem is proved. \square

Similarly, if $\text{co}(G)$ is dense in the Hilbert space H , or if the target functions are located within $\text{co}(G)$, we can infer that $m = 0$. This leads to the following corollary:

Corollary 4.3. Assume the inputs $\{\mathbf{x}_i, i = 1, 2, \dots, K\}$ generated with a uniform probability distribution, where K is the number of training samples. If $\text{co}(G)$ is dense in Hilbert space H or target functions located in $\text{co}(G)$, i.e., $f \in \text{co}(G)$, then for every $c > (p+1)B$ and $p > -1$, there exists a sequence $f_n = \alpha_n f_{n-1} + \bar{\alpha}_n g_n$ such that

$$\lim_{n, K \rightarrow \infty} \left| \sum_{i=1}^K (f_n(\mathbf{x}_i) - f(\mathbf{x}_i))^2 \right| = 0$$

where $\alpha_n = \frac{n-1}{n+p}$, $\bar{\alpha}_n = \frac{p+1}{n+p}$ and $\|g\| \leq B$.

Remark 4: Theorem 4.2 and Corollary 4.3 provide discrete convergent descriptions for the case where input data satisfies a uniform distribution, thus allowing for the construction of the entire network using limiting input data. However, it should be noted that this assumption is often not satisfied in practical scenarios due to input uncertainty. Nonetheless, numerous simulation results demonstrate that the networks constructed using our proposed methods can achieve good performance despite input uncertainty, thus providing evidence that our theories are robust. In the following section, we will present our algorithm based on these theories and demonstrate its effectiveness through experimental results.

5. Algorithm Implementations

Based on the theories presented above, the entire network is constructed using a generalized convex iteration. For each new added hidden neuron, it is necessary to find the minimum residual error $\inf_{g \in \mathcal{G}} \|\alpha_n F + \bar{\alpha} g - f\|^2$. There are two methods to obtain this minimum value. One is to use the traditional backpropagation (BP) algorithm to find the optimal weights for g_n . The other is to approximate the optimal weights by calculating the residual error for randomly generated group weights. In the following sections, we will provide detailed descriptions of the implementation of both methods.

5.1. Algorithm Based On Back Propagation(BP)

In this section, we will utilize the backpropagation (BP) algorithm to update the network weights. For a newly added neuron g_n , we first freeze the weights of the previous hidden neurons $g_i, i = 1, \dots, n-1$. The network is then constructed using our proposed convex iteration. Next, input data is used to obtain predicted outputs. Finally, a loss function is constructed by comparing the predicted outputs and the actual outputs. Since the previous weights have been frozen, the BP algorithm only updates the newly added neuron g_n . The entire pseudo-code for this process is described as follows:

5.2. Algorithm Based On Random Searching

In this section, we propose an alternative method to obtain the minimum residual error, which addresses the complexity of the backpropagation (BP) algorithm. For a newly added neuron g_n , a group of weights \mathbf{w}_n^i and b_n^i are randomly generated, where $i = 1, \dots, K$ and $K > 1$ is a hyperparameter. The larger the value of K , the more likely we are to approximate the minimum residual error. The following is the pseudo-code for the algorithm:

When comparing the above methods, it can be seen that our proposed random searching method is superior to the traditional backpropagation (BP) algorithm. This is due to the fact that BP may cause the entire network to converge to a local minimum, and the issues of vanishing and exploding gradients can make the network difficult to train. Our random searching method addresses these shortcomings, making it a more effective method for network training.

6. Experiments

In this section, we apply our algorithm to a variety of regression problems in order to verify our conclusions. However, due to the length constraints of the conference paper, we have

Algorithm 1 BP Algorithm Implementation

For the training data $\{(\mathbf{x}_i, y_i)_{i=1, \dots, K}\}$, we preset several super parameters: the last expected accuracy ϵ , the early stop maximum hidden neuron number n_{\max} , any bounded nonlinear activation function g , trial times k and any $p > -1$:

- (1) **Initialization:** Set the beginning hidden neuron number as $n = 0$, the initial error $E = [y_1, \dots, y_K]^T$ is the original data output, and $F = [f_1, \dots, f_K]^T$ is the target vector of the target function and $G = [g(1), \dots, g(K)]^T$ is the activation vector of the new neuron for all the K training samples.
- (2) **Learning step:**
while $n < n_{\max}$ and $\|E\| > \epsilon$
 - (a) Add one new hidden neuron: $n = n + 1$.
 - (b) Based Theorem 3.1, set the output weight $\alpha_n = \frac{n-1}{n+p}$ and recalculate all existing hidden neuron weights:

$$\alpha_i = \alpha_n \cdot \alpha_i, \quad i = 1, \dots, n-1$$

- (c) Frozen previous hidden neurons.
- (d) Finding new optimal neuron weights
- (e) Randomly generate new added neuron weights (\mathbf{w}_n, b_n) and update G .
- (f) Construct loss function $loss(\mathbf{w}_n, b_n)$

$$loss(\mathbf{w}_n, b_n) = \sum_{i=1}^K |\alpha_n f_i + (1 - \alpha_n) g_i - y_i|$$

- (g) Use BP algorithm to get optimal weights (\mathbf{w}_n, b_n) and update G .
- (h) update $E = \alpha_n E + (1 - \alpha_n)(F - G)$

endwhile

Algorithm 2 Random Searching Algorithm Implementation

For the training data $\{(\mathbf{x}_i, y_i)_{i=1, \dots, K}\}$, we preset several super parameters: the last expected accuracy ϵ , the early stop maximum hidden neuron number n_{\max} , any bounded nonlinear activation function g , trial times k and any $p > -1$:

- (1) **Initialization:** Set the beginning hidden neuron number as $n = 0$, the initial error $E = [y_1, \dots, y_K]^T$ is the original data output, and $F = [y_1, \dots, y_K]^T$ is the target vector of the target function and $G = [g(1), \dots, g(K)]^T$ is the activation vector of the new neuron for all the K training samples.
- (2) **Learning step:**
while $n < n_{\max}$ and $\|E\| > \epsilon$
 - (a) Add one new hidden neuron: $n = n + 1$.
 - (b) Based Theorem 3.1, set the output weight $\alpha_n = \frac{n-1}{n+p}$ and recalculate all existing hidden neuron weights:

$$\alpha_i = \alpha_n \cdot \alpha_i, \quad i = 1, \dots, n-1$$

- (c) For $i = 1 : k$
 - (i) Randomly create neuron weights $(\mathbf{w}_n^{(i)}, b_n^{(i)})$.
 - (ii) Calculate every error $E^{(i)}$

$$E^{(i)} = \alpha_n E + (1 - \alpha_n)(F - G^{(i)})$$

- (d) end For
 - (e) Let $i^* = \{i \mid \min_{1 \leq i \leq k} \|E^{(i)}\|\}$. Set $E = E^{(i^*)}$, $\mathbf{w}_n = \mathbf{w}_n^{(i^*)} L$ and $b_n = b_n^{(i^*)}$.
- endwhile
-

only selected 10 regression datasets to demonstrate the performance of our algorithm. Table 1 provides descriptions of the 10 datasets that have been selected. All the simulations were run in a MATLAB environment without GPU acceleration, as the focus of this paper is to demonstrate the correctness of our proposed theory.

| Name | Data | | Attributes |
|-------------------|----------|---------|------------|
| | Training | Testing | |
| Abalone | 2000 | 2177 | 8 |
| Ailerons | 7154 | 6596 | 39 |
| Airplane | 450 | 500 | 9 |
| Bank | 4500 | 3692 | 8 |
| Boston | 250 | 256 | 13 |
| California | 8000 | 12640 | 8 |
| Computer Activity | 4000 | 4192 | 12 |
| Delta Ailerons | 3000 | 4129 | 5 |
| Delta Elevators | 4000 | 5517 | 6 |
| Kinematics | 4000 | 4192 | 8 |

Table 1. Descriptions of 10 Selected Regression Datasets

In our experiments, all inputs were normalized to the range $[-1, 1]$, and the outputs were transformed to $[0, 1]$. The input weights \mathbf{a}_i and biases b_i were randomly generated from the range $[-1, 1]$. Our target termination error was set to $\epsilon = 0.001$, and the maximum number of hidden neurons was 1000. For each result, we recorded 30 times and calculated the average value as the final result, which demonstrates the stability and robustness of our experiments.

| Name | $p = -0.5$ | | | | $p = 0.5$ | | | |
|-----------------|------------|--------|--------|--------|-----------|--------|--------|--------|
| | 100th | 200th | 500th | 1000th | 100th | 200th | 500th | 1000th |
| Abalone | 0.0925 | 0.0919 | 0.0914 | 0.0912 | 0.0922 | 0.0915 | 0.0911 | 0.0909 |
| Ailerons | 0.0802 | 0.0792 | 0.0784 | 0.0783 | 0.0799 | 0.0791 | 0.0786 | 0.0784 |
| Airplane | 0.1515 | 0.1505 | 0.1492 | 0.1488 | 0.1513 | 0.1503 | 0.1494 | 0.1493 |
| Bank | 0.1723 | 0.1715 | 0.1711 | 0.1709 | 0.1716 | 0.1715 | 0.1713 | 0.1711 |
| Boston | 0.1461 | 0.1451 | 0.1461 | 0.1451 | 0.1467 | 0.1453 | 0.1448 | 0.1443 |
| California | 0.2089 | 0.2084 | 0.2079 | 0.2078 | 0.2081 | 0.2079 | 0.2071 | 0.2070 |
| Computer | 0.1562 | 0.1561 | 0.1558 | 0.1556 | 0.1546 | 0.1543 | 0.1539 | 0.1538 |
| Delta Ailerons | 0.0483 | 0.0479 | 0.0475 | 0.0473 | 0.0495 | 0.0488 | 0.0484 | 0.0482 |
| Delta Elevators | 0.0621 | 0.0616 | 0.0611 | 0.0609 | 0.0636 | 0.0629 | 0.0624 | 0.0622 |
| Kinematics | 0.1453 | 0.1445 | 0.1439 | 0.1437 | 0.1448 | 0.1440 | 0.1435 | 0.1434 |

Table 2. Testing mean squared error comparison of different neurons (100th, 200th, 500th, 1000th) under different $p = -0.5, 0.5$ in Sigmoid activation function

The mean squared errors for different numbers of neurons (100th, 200th, 500th, 1000th) under different $p = -0.5, 0.5$ values in the Sigmoid activation function are presented in Table 2. The results indicate that the testing errors decrease as the number of neurons increases. Even for the negative parameter value $p = -0.5$, our algorithm still exhibits a stable downward trend, which is in agreement with Theorem 3.1. To further reinforce the validity of our results, we also used Gaussian activation functions with different $p = -0.2, 0.8$ values for the same numbers of neurons (100th, 200th, 500th, 1000th) in Table 3. The results are consistent with the aforementioned analysis.

Figure 1 illustrates the testing mean squared error (MSE) curves as a function of the number of neurons for the California dataset. To differentiate from previous convex iterations,

| Name | $p = -0.2$ | | | | $p = 0.8$ | | | |
|-----------------|------------|--------|--------|--------|-----------|--------|--------|--------|
| | 100th | 200th | 500th | 1000th | 100th | 200th | 500th | 1000th |
| Abalone | 0.0944 | 0.0938 | 0.0933 | 0.0930 | 0.0948 | 0.0941 | 0.0935 | 0.0933 |
| Ailerons | 0.3951 | 0.3832 | 0.3919 | 0.3925 | 0.3914 | 0.3926 | 0.3899 | 0.3910 |
| Airplane | 0.1847 | 0.1835 | 0.1831 | 0.1826 | 0.1844 | 0.1828 | 0.1816 | 0.1815 |
| Bank | 0.1657 | 0.1653 | 0.1652 | 0.1651 | 0.1653 | 0.1651 | 0.1649 | 0.1649 |
| Boston | 0.1866 | 0.1858 | 0.1852 | 0.1849 | 0.1900 | 0.1887 | 0.1883 | 0.1880 |
| California | 0.2178 | 0.2173 | 0.2169 | 0.2167 | 0.2179 | 0.2174 | 0.2169 | 0.2168 |
| Computer | 0.2831 | 0.2817 | 0.2848 | 0.2851 | 0.2861 | 0.2856 | 0.2845 | 0.2847 |
| Delta Ailerons | 0.0571 | 0.0562 | 0.0553 | 0.0548 | 0.0574 | 0.0563 | 0.0550 | 0.0545 |
| Delta Elevators | 0.0734 | 0.0723 | 0.0714 | 0.0711 | 0.0720 | 0.0708 | 0.0696 | 0.0690 |
| Kinematics | 0.1608 | 0.1595 | 0.1588 | 0.1585 | 0.1606 | 0.1594 | 0.1586 | 0.1583 |

Table 3. Testing mean squared error comparison of different neurons (100th, 200th, 500th, 1000th) under different $p = -0.2, 0.8$ in Gaussian activation function

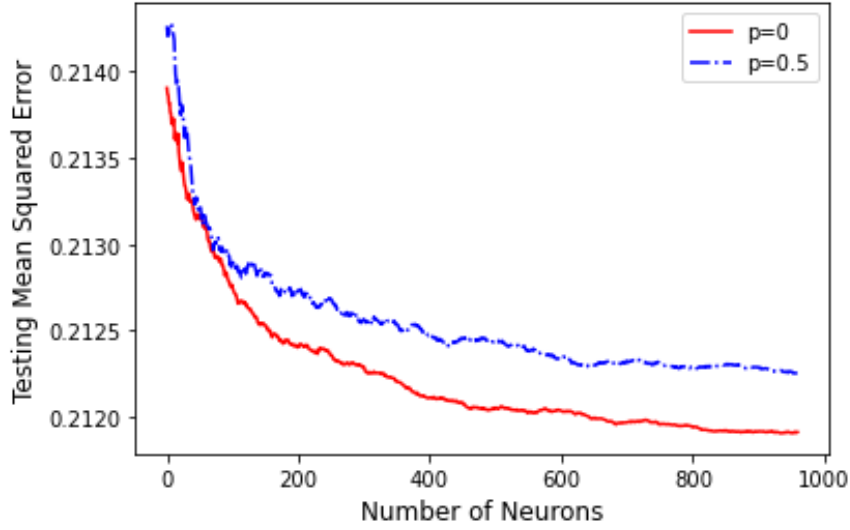


Figure 1. Testing mean squared errors with different p for California case.

we selected $p = 0, 0.5$ and used Gaussian activation functions. The results in Fig. 1 demonstrate that the MSE curves for different p values exhibit a consistent decreasing trend, which confirms the validity of our convergence theories.

For the sake of simplicity, all the following simulations use $p = 0$ and the sigmoid activation function to describe our results. Fig.2 shows the training and testing MSE curves with the growth of the Sigmoid neuron number. Seen from Fig.2, our algorithm can achieve a better generalization performance with the growth of neurons, which verifies our approximation results.

The above paragraph describes the results of an experimental evaluation of the proposed algorithm using the sigmoid activation function and $p = 0$. Fig.2 illustrates the training and testing mean squared error (MSE) as a function of the number of sigmoid neurons used. The results demonstrate that the algorithm can achieve improved generalization performance as the number of neurons increases, which confirms the validity of the approximation results presented in the paper.

The results presented in Fig.2 and Fig.3 demonstrate the effectiveness of the proposed al-

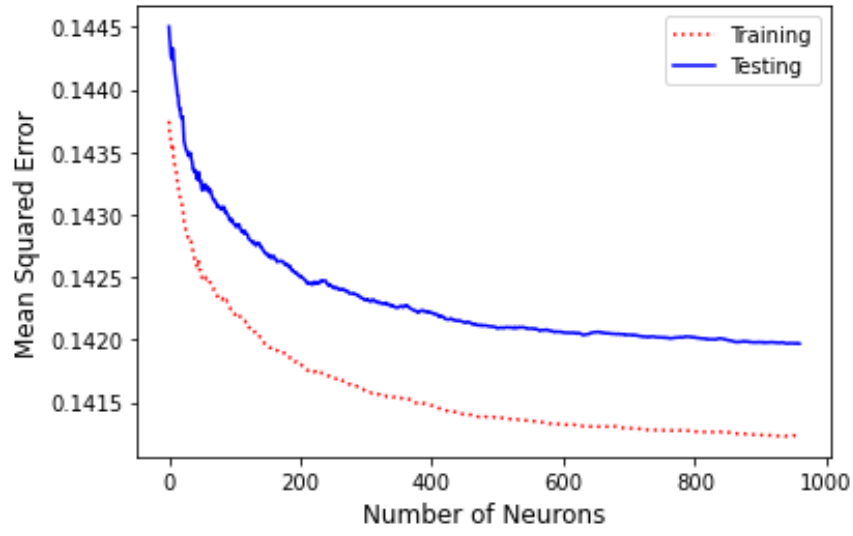


Figure 2. Training and testing mean squared errors for Kinematics case.

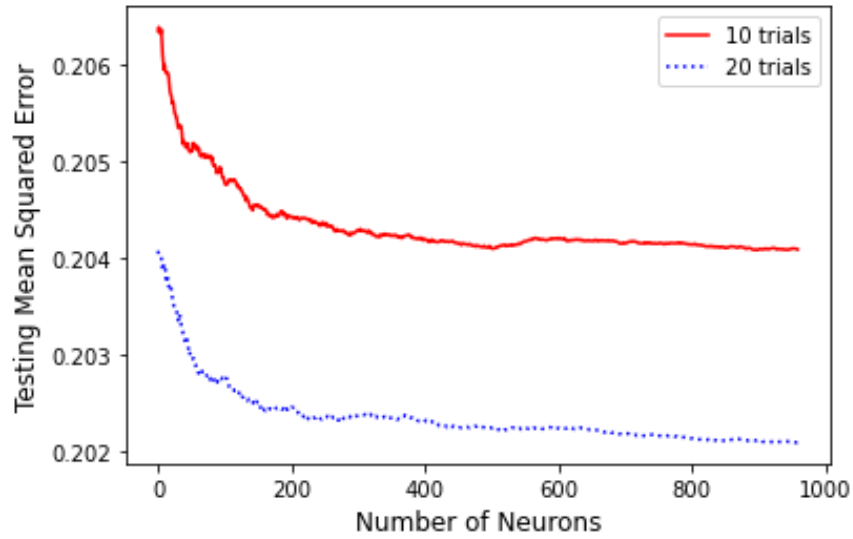


Figure 3. Testing mean squared error with different k for California case.

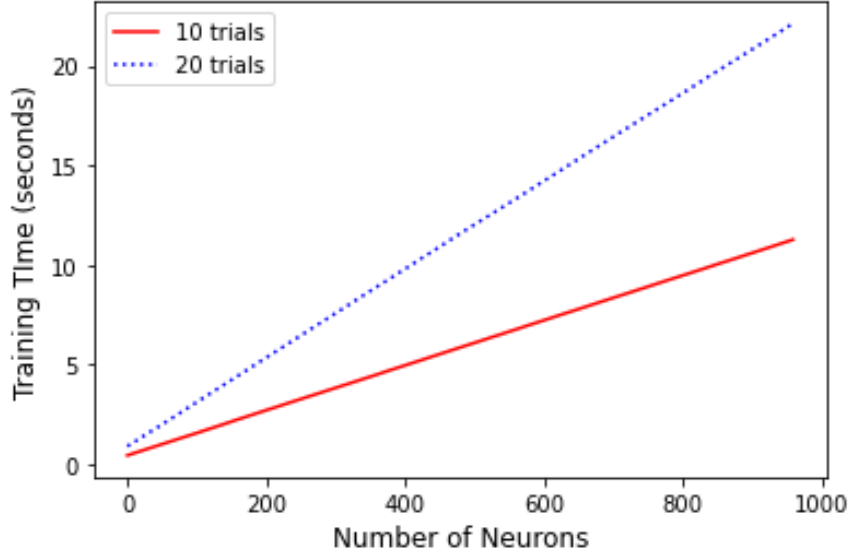


Figure 4. Training time comparison with different k for California case.

gorithm in achieving better generalization performance as the number of neurons increases, which confirms the validity of the approximation theories presented in this paper. In particular, Fig.3 illustrates that increasing the number of trial choices in the random searching method leads to improved approximation performance. However, it should be noted that this improvement comes at the cost of increased training time, as shown in Fig.4. This figure illustrates that the training time increases linearly with the number of trial choices, highlighting the trade-off between approximation performance and computational cost.

7. Conclusion

In this paper, a generalized convex incremental structure is proposed which offers a greater diversity in network design and maintains the same convergence rate. Additionally, our theory also provides indirect evidence for the universal approximation capability of neural networks when certain conditions are met. Due to the unknown nature of objective functions and the incompleteness of inputs, it can be challenging to verify our theory. To address this, we introduced the concept of discrete approximation and the use of random searching methods for algorithm implementation. Although our implementation may not be mathematically precise, our experimental results demonstrate the robustness and good generalization of our algorithm, as demonstrated on several selected regression datasets. We hope that our research will draw attention to the importance of theoretical research in neural networks.

Disclosure Statement

All authors have declared that: (i) no support, financial or otherwise, has been received from any organization that may have an interest in the submitted work; and (ii) there are no other relationships or activities that could appear to have influenced the submitted work.

Data Availability

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

References

- [1] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: NIPS; 2012.
- [2] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. In: Computer Science; 2012.
- [3] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: CVPR; 2016.
- [4] Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators. *Neural Networks*. 1989;2:359–366.
- [5] Cybenko G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*. 1989;2:303–314.
- [6] Ito Y. Approximation of functions on a compact set by finite sums of a sigmoid function without scaling. *Neural Networks*. 1991;4:817–826.
- [7] Hornik K. Approximation capabilities of multilayer feedforward networks. *Neural Networks*. 1991;4:251–257.
- [8] Hornik K. Some new results on neural network approximation. *Neural Networks*. 1993;6:1069–1072.
- [9] Leshno M, Lin VY, Pinkus A, et al. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*. 1993;6:861–867.
- [10] Chen T, Chen H, Liu RW. Approximation capability in $C(\mathbf{R}^n)$ by multilayer feedforward networks and related problems. *IEEE Transactions on Neural Networks*. 1995;6(1):25–30.
- [11] Chen T, Chen H. Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks. *IEEE Transactions on Neural Networks*. 1995;6(4):904–910.
- [12] Lee WS, Bartlett PL, Williamson RC. Efficient agnostic learning of neural networks with bounded fan-in. *IEEE Transactions on Information Theory*. 1996;42(6):2118–2132.
- [13] Maierov V, Pinkus A. Lower bounds for approximation by mlp neural networks. *Neurocomputing*. 1999;25:81–91.
- [14] Meir R, Maierov VE. On the optimality of neural-network approximation using incremental algorithms. *IEEE Transactions on Neural Networks*. 2000;11(2):323–337.
- [15] Lavretsky E. On the geometric convergence of neural approximations. *IEEE Transactions on Neural Networks*. 2002;13(2):274–282.
- [16] Xiang C, Shenqiang, Lee TH. Geometrical interpretation and architecture selection of mlp. *IEEE Transactions on Neural Networks*. 2005;16(1):84–96.
- [17] Huang GB, Chen L, Siew CK. Universal approximation using incremental constructive feed-forward networks with random hidden nodes. *IEEE Transactions On Neural Networks*. 2006; 17(4):879–892.
- [18] Huang GB, Chen L. Convex incremental extreme learning machine. *Neurocomputing*. 2007; 70:3056–3062.
- [19] Jones LK. A simple lemma on greedy approximation in hilbert space and convergence rates for projection pursuit regression and neural networks. *The Annals of Statistics*. 1992;20(1):608–613.
- [20] Barron AR. Universal approximation bounds for superpositions of a sigmoid function. *IEEE Transactions on Information Theory*. 1993;39(3):930–945.
- [21] Koiran P. Efficient learning of continuous neural networks. In: *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory*; New Brunswick, New Jersey; 1994. p. 348–355.
- [22] Chen L, Huang GB, Pung HK. Systemical convergence rate analysis of convex incremental feed-

- forward neural networks. *Neurocomputing*. 2009;72:2627–2635.
- [23] Kaminski W, Strumillo P. Kernel orthonormalization in radial basis function neural networks. *IEEE Transactions On Neural Networks*. 1997;8(5):1177–1183.