



Article

Neural Networks with Fractal Architecture

Alireza Khalili Golmankhaneh ^{1,*} , Cristina Serpa ^{2,3,4,*} , Rawid Banchuin ⁵ and Palle E. T. Jørgensen ⁶ ¹ Department of Physics, Ur.C., Islamic Azad University, Urmia 63896, West Azerbaijan, Iran² Department of Exact Sciences and Engineering, Military Academy, Portuguese Army, 1169-203 Lisbon, Portugal³ CINAMIL—Military Academy Research Center, 1169-203 Lisbon, Portugal⁴ CEMS.UL—Center for Mathematical Studies, 1749-016 Lisbon, Portugal⁵ Faculty of Engineering and Graduate School of Information, Technology, Siam University, 38 Petchkasem Road, Bangwa, Phasicharoen, Bangkok 10160, Thailand⁶ Department of Mathematics, The University of Iowa, Iowa City, IA 52242-1419, USA

* Correspondence: alireza@siam.edu (A.K.G.); mcserpa@fc.ul.pt (C.S.); Tel.: +351-213-186-900 (C.S.)

Abstract

In this paper, we propose the Fractal Architecture Neural Network (FANN), a recursive neural framework inspired by self-similar fractal geometry. The architecture is governed by a fractal dimension parameter α , which controls the branching structure and connectivity density of the network, enabling multiscale feature representation through parameter sharing across recursive paths. We evaluate FANN on synthetic nonlinear regression tasks and compare it with a standard artificial neural network (ANN) and FractalNet in terms of accuracy, training behavior, and model complexity. Experimental results show that FANN achieves competitive or improved predictive performance under comparable computational budgets, demonstrating effective accuracy-to-parameter efficiency. These results suggest that fractal-inspired recursive connectivity can provide a compact mechanism for hierarchical representation learning in neural networks.

Keywords: fractal neural networks; self-similarity; fractal dimension; FractalNet model

1. Introduction

Fractal geometry, introduced by Benoît Mandelbrot [1], studies structures whose complexity exceeds that suggested by their topological dimensions [2,3]. Building on these ideas, fractal calculus extends classical calculus to non-smooth, self-similar sets and provides mathematical tools for analyzing irregular structures such as the Cantor and Koch sets [4]. These developments have inspired the application of fractal concepts in various scientific fields where hierarchical and multiscale phenomena naturally arise.

In parallel, ANNs have achieved remarkable success across a wide range of applications, including computer vision, speech recognition, and natural language processing [5–13]. Despite their expressive power, conventional neural networks often rely on large numbers of parameters and deep architectures to capture complex patterns. Such designs may increase computational cost and may not explicitly exploit hierarchical multiscale structures present in many types of data [14]. Residual learning frameworks have been introduced to ease the training of substantially deeper neural networks. Comprehensive empirical evidence has been provided to show that deep residual networks are easier to optimize and achieve higher accuracy [15]. Dense Convolutional Networks (DenseNets) have been introduced to improve feature propagation and reduce the number of parameters in deep networks. Significant improvements over state-of-the-art methods



Academic Editors: Astero Provata and António Lopes

Received: 5 May 2026

Revised: 5 June 2026

Accepted: 26 June 2026

Published: 30 June 2026

Copyright: © 2026 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 \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

have been achieved on several object recognition benchmark tasks using DenseNets [16]. A design strategy for neural network macro-architecture based on self-similarity has been introduced through fractal network structures. High-performance fixed-depth subnetworks have been extracted using drop-path regularization in fractal architectures [17]. A clustering-based regularization method has been proposed to facilitate generalization and interpretability in deep networks. The effectiveness of the proposed approach has been demonstrated on tasks such as unsupervised learning, classification, and zero-shot learning [18]. A new family of deep neural network models has been introduced using continuous-depth architectures based on differential equations. Scalable backpropagation through arbitrary ODE solvers has been demonstrated to enable end-to-end training within larger models [19]. Interestingly, studies of biological neural systems indicate that neural connectivity often exhibits fractal-like organization. The human brain, for example, displays self-similar connectivity patterns across multiple spatial scales [20,21]. These structural properties are believed to support efficient information processing, robustness, and adaptive learning capabilities [22,23]. Such observations motivate the exploration of neural architectures that incorporate fractal principles into their structural design.

Motivated by these ideas, fractal-inspired neural architectures have been proposed in recent years. In particular, the FractalNet model demonstrates how recursive and self-similar structures can be used to construct deep networks with multiple computational paths [17]. In addition, recent studies suggest that the connectivity patterns of certain neural systems exhibit fractal characteristics, highlighting the relationship between architectural complexity and representational capacity [24].

In this work, we introduce the FANN, a biologically inspired framework in which each layer is recursively composed of smaller sublayers, forming a hierarchical self-similar topology. The architecture is governed by a structural parameter that controls the effective fractal dimension of the network, allowing the density and scaling of connections to be adjusted. Through this design, the proposed architecture provides a mechanism for constructing multiscale representations while maintaining a compact parameterization.

To investigate the properties of the proposed model, a series of experiments are conducted on nonlinear regression tasks. The study includes a comparative analysis between the proposed FANN architecture, a conventional feedforward ANN, and the fractal-inspired FractalNet architecture under comparable parameter settings. In addition, the influence of the fractal dimension on the learning behavior and approximation capability of the network is examined.

These investigations aim to provide insight into how fractal-inspired architectural principles can be incorporated into neural network design and how such structures influence the learning dynamics of the resulting models.

The remainder of this paper is organized as follows.

In Section 2, we introduce the proposed fractal artificial neural network architecture and describe how its fractal dimension is defined. In Section 3, we present the theoretical analysis of the proposed framework. A detailed comparison between FANN, ANN, and FractalNet is provided in Section 4, particularly in the CIFAR-10 experimental results.

Finally, Section 5 concludes the paper and discusses the main findings.

2. Fractal Architecture Neural Networks

In this section, we introduce Fractal Architecture Neural Networks (FANNs), a class of neural architectures constructed through the recursive, self-similar composition of a fundamental neural module. The key idea is to impose a multiscale hierarchical connectivity pattern within the computational graph, enabling a structural investigation of how recursive topology influences representation capacity and information flow.

To highlight the structural differences between conventional neural architectures and the proposed model, Figure 1 provides a comparison between a standard feedforward ANN and the proposed FANN.

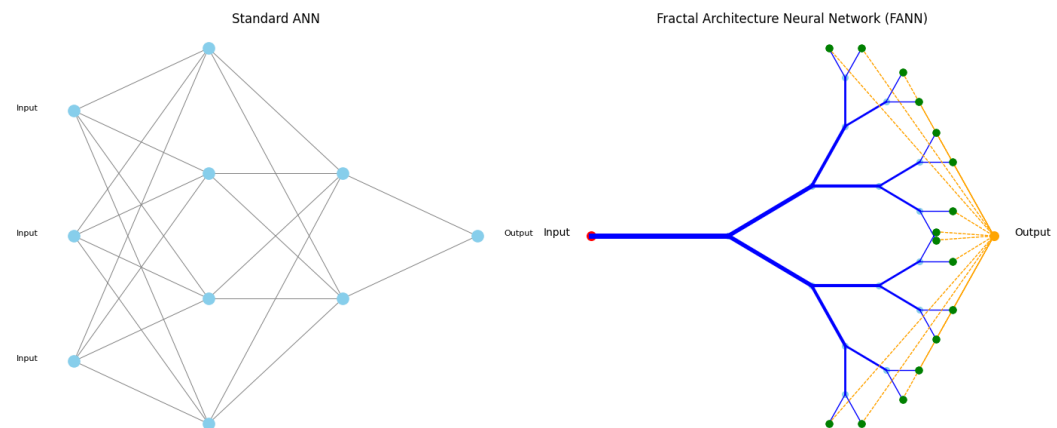


Figure 1. Comparison between a conventional feedforward artificial neural network (ANN) and the proposed Fractal Architecture Neural Network (FANN). The ANN follows a sequential layered topology with fully connected transformations, whereas the FANN expands recursively through self-similar branching that generates multiple hierarchical signal propagation paths. Nonlinear activations (ReLU) are applied only at the leaf nodes, and their outputs are aggregated to produce the final network output.

In a conventional ANN, neurons are organized in sequential layers, and information propagates through fully connected transformations from the input layer to the output layer. Model capacity is typically increased by enlarging the network depth or width, while the connectivity pattern remains strictly layered.

In contrast, the proposed FANN introduces a recursive branching structure inspired by fractal self-similarity. Instead of stacking layers sequentially, the architecture expands by repeatedly duplicating a basic neural module, producing a hierarchy of interconnected branches. This recursive organization generates multiple signal propagation paths between the input and output, enabling multiscale information processing.

At each recursive level, linear transformations are duplicated across branches, and their outputs are subsequently aggregated. Nonlinear activations (ReLU) are applied only at the terminal (leaf) nodes, allowing the effect of recursive linear expansion to be separated from nonlinear transformation. A scale factor s controls the contribution of each branch prior to aggregation, thereby regulating the effective structural complexity of the architecture.

2.1. Fractal Dimension of the Architecture

To quantify the recursive structure of the proposed Fractal Architecture Neural Network (FANN), we define an effective fractal dimension parameter α .

Let $b \geq 2$ denote the branching factor and let $s \in (0, 1)$ denote the self-similarity scaling coefficient. The effective fractal dimension is defined as

$$\alpha = \frac{\log b}{-\log s}. \quad (1)$$

This expression is consistent with the classical box-counting dimension of self-similar structures, where the equilibrium relation

$$b s^\alpha = 1 \quad (2)$$

holds.

In the proposed framework, α represents an effective dimension of the computational graph induced by recursive branching, rather than the geometric dimension of a Euclidean fractal set. It characterizes how rapidly the number of active computational paths grows relative to the contraction induced by recursive scaling.

For tree-structured recursive networks operating in a stable hierarchical regime, the parameters are typically chosen such that the branching growth and contraction satisfy

$$bs > 1 \quad \text{and} \quad bs^2 < 1, \quad (3)$$

which ensures a dense but non-space-filling propagation structure. Under these conditions, the effective dimension satisfies

$$1 < \alpha < 2, \quad (4)$$

corresponding to a balanced recursive tree regime where information propagation is sufficiently rich for multiscale representation while remaining computationally controlled.

Outside this regime, α remains a positive continuous parameter determined by (b, s) , and may lie outside $(1, 2)$ depending on the degree of sparsity or density of the recursive construction. Larger values of b increase α , while stronger contraction (smaller s) reduces it, thereby directly controlling the effective connectivity of the hierarchical architecture.

Example 1. To illustrate the interpretation of the fractal dimension parameter α , we consider a recursive fractal tree generated through self-similar branching. The effective fractal dimension is estimated numerically using the box-counting method.

The box-counting dimension satisfies

$$N(\epsilon) \sim \epsilon^{-\alpha}, \quad (5)$$

where $N(\epsilon)$ denotes the number of occupied boxes of size ϵ . Taking logarithms yields

$$\log N(\epsilon) = \alpha \log \left(\frac{1}{\epsilon} \right). \quad (6)$$

Thus, the fractal dimension is obtained as the slope of the corresponding log–log plot.

In the numerical experiment, the generated fractal tree is analyzed using box sizes

$$\epsilon \in \{4, 8, 16, 32, 64\},$$

yielding an estimated box-counting dimension

$$\alpha \approx 1.98. \quad (7)$$

Figure 2 illustrates the numerical estimation of the effective fractal dimension used in the proposed FANN architecture.

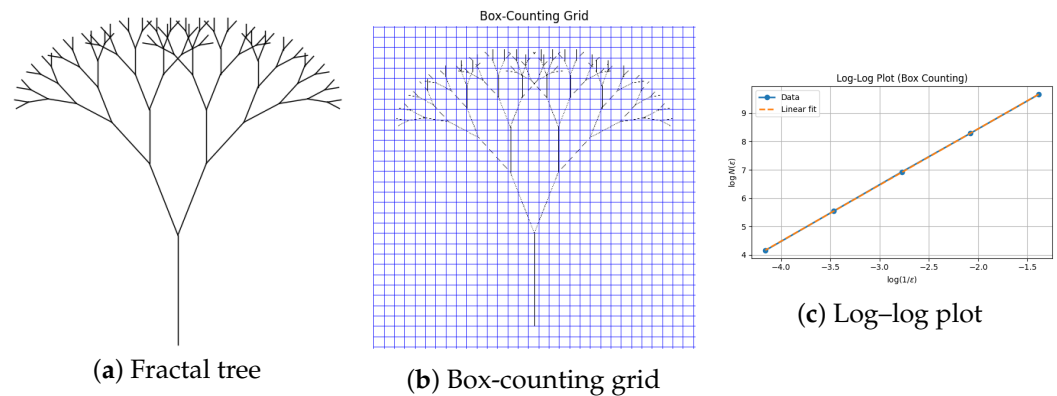


Figure 2. Box-counting estimation of the fractal dimension for a recursive fractal tree. The slope of the log–log regression provides the effective box-counting dimension.

2.2. Recursive Fractal Architecture

To formally define the recursive structure of the proposed FANN, we introduce a hierarchy of operator-valued mappings indexed by recursion depth d . Let $x \in \mathbb{R}^n$ denote the input vector, and let

$$\sigma : \mathbb{R} \rightarrow \mathbb{R}$$

be a pointwise nonlinear activation function.

Base module ($d = 0$).

The initial embedding operator is defined as

$$\mathcal{F}^{(0)}(x) = \sigma(W^{(0)}x + b^{(0)}), \quad (8)$$

where $W^{(0)} \in \mathbb{R}^{m \times n}$ and $b^{(0)} \in \mathbb{R}^m$. This induces the mapping $\mathcal{F}^{(0)} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, which serves as the initial feature representation.

Recursive branching operator.

For $d \geq 0$, we define a set of branch transformations $\{\mathcal{T}_k^{(d)}\}_{k=1}^b$ acting on $\mathbb{R}^m$. Each $\mathcal{T}_k^{(d)}$ represents a parameter-shared or partially shared transformation applied along the k -th branch.

The recursive evolution is defined as

$$\mathcal{F}^{(d+1)}(x) = s \sum_{k=1}^b \mathcal{T}_k^{(d)}(\mathcal{F}^{(d)}(x)), \quad (9)$$

where $b \geq 2$ is the branching factor and $s \in (0, 1)$ is a scaling coefficient controlling aggregation magnitude.

A standard instantiation is

$$\mathcal{T}_k^{(d)}(z) = W_k^{(d)}z + b_k^{(d)},$$

with optional weight sharing $W_k^{(d)} \equiv W^{(d)}$ across branches.

Unrolled multiscale representation.

Iterating Equation (9) yields the following expansion:

$$\mathcal{F}^{(D)}(x) = s^D \sum_{(k_1, \dots, k_D) \in \{1, \dots, b\}^D} \left(\mathcal{T}_{k_D}^{(D-1)} \circ \dots \circ \mathcal{T}_{k_1}^{(0)} \right) (\mathcal{F}^{(0)}(x)). \quad (10)$$

Each multi-index $(k_1, \dots, k_D)$ corresponds to a unique computational path in a depth- D branching tree.

Stability and scaling behavior.

A sufficient condition for asymptotic contraction of the recursive mapping is $bs < 1$, which ensures that the operator norm of the forward propagation decreases with depth.

In practical implementations, the model may operate in a controlled non-contractive regime ($bs \gtrsim 1$), where expressivity is increased. In this case, stability is enforced empirically through bounded-depth recursion, normalization, and initialization strategies (e.g., modified He initialization), ensuring that activations remain numerically stable during training.

Interpretation.

The mapping $\mathcal{F}^{(d)}$ defines a self-similar branching dynamical system. Each recursion depth induces a multiscale decomposition:

- Small d : coarse-grained feature extraction;
- Large d : fine-grained hierarchical refinement;
- Full depth D : aggregation over exponentially many computational paths.

Final output.

The network output is defined as

$$y(x) = \mathcal{F}^{(D)}(x), \quad (11)$$

representing the final multiscale representation produced by the recursive fractal architecture.

2.3. Connectivity Density and Fractal Scaling

To make the recursive connectivity structure reproducible, we define a binary connectivity mask associated with each recursion level d .

Let

$$M^{(d)} \in \{0, 1\}^{m \times m}$$

denote the connectivity matrix at depth d , where

$$M_{ij}^{(d)} = \begin{cases} 1, & \text{if a connection exists between neurons } i \text{ and } j, \\ 0, & \text{otherwise.} \end{cases} \quad (12)$$

The effective connectivity density at recursion depth d is defined by

$$\rho_d = \frac{1}{m^2} \sum_{i,j} M_{ij}^{(d)}, \quad (13)$$

where m denotes the layer width.

To incorporate fractal scaling into the architecture, we assume that the connectivity density decreases recursively according to the scaling coefficient $s \in (0, 1)$:

$$\rho_d = \rho_0 s^d, \quad (14)$$

where ρ_0 is the initial connectivity density.

Using the relation

$$\alpha = \frac{\log b}{-\log s}, \quad (15)$$

the recursive scaling factor can be written as

$$s = b^{-1/\alpha}. \quad (16)$$

Therefore, the connectivity density becomes

$$\rho_d = \rho_0 b^{-d/\alpha}. \quad (17)$$

This expression explicitly shows how the fractal dimension parameter α regulates the density of recursive connections within the computational graph. Larger values of α produce denser recursive connectivity patterns, while smaller values yield sparser architectures with reduced parameter interactions.

2.4. Multiscale Representation

The recursive structure of the proposed FANN naturally generates feature representations at multiple scales. Each recursion depth d is interpreted as a distinct computational scale within the network hierarchy.

For recursion depth d , the scale-indexed feature map is defined by

$$h^{(d)}(x) = \mathcal{F}^{(d)}(x), \quad (18)$$

where $h^{(d)}(x) \in \mathbb{R}^m$ denotes the feature representation extracted at scale d .

Smaller values of d correspond to coarse representations with limited recursive refinement, while larger values of d produce finer multiscale features through deeper recursive aggregation.

The final multiscale representation is obtained by combining all recursive scales:

$$H(x) = \sum_{d=0}^D \omega_d h^{(d)}(x), \quad (19)$$

where

- D is the maximum recursion depth;
- $\omega_d \in \mathbb{R}$ denotes the aggregation weight associated with scale d ;
- $H(x)$ is the global multiscale representation.

Equation (19) shows that the network output integrates hierarchical information across multiple recursive scales. Consequently, the FANN architecture captures both coarse and fine structural features while preserving recursive self-similarity within the computational graph.

2.5. Algorithmic Implementation of FANN

This section provides a concise and reproducible description of the proposed Fractal Architecture Neural Network (FANN), connecting the mathematical formulation in Equations (8)–(10) to its computational implementation.

The model is defined as a recursive operator system of depth D , where each level applies a structurally identical transformation with shared parameters, ensuring self-similarity and parameter efficiency.

2.5.1. Base Module

The base transformation is given by Equation (8):

$$\mathcal{F}^{(0)}(x) = \sigma(W^{(0)}x + b^{(0)}), \quad (20)$$

which serves as the initial feature extractor.

2.5.2. Recursive Computation

The recursive operator in Equation (9) is implemented via the FANNBlock:

$$\mathcal{F}^{(d+1)}(x) = \sum_{k=1}^b s \cdot \mathcal{T}_k^{(d)}(\mathcal{F}^{(d)}(x)), \quad (21)$$

where $b \geq 2$ is the branching factor and $s \in (0, 1)$ is the contraction coefficient.

At each depth, the computation is:

$$y_1 = \mathcal{F}^{(d-1)}(x), \quad y_2 = \mathcal{F}^{(d-1)}(x), \quad (22)$$

$$a_1 = \text{Attention}(y_1), \quad a_2 = \text{Attention}(y_2), \quad (23)$$

$$y = a_1 \odot y_1 + a_2 \odot y_2, \quad (24)$$

$$\mathcal{F}^{(d)}(x) = \phi(y), \quad (25)$$

where $\phi(\cdot)$ is a fusion operator.

2.5.3. Forward Propagation

$$y(x) = \mathcal{F}^{(D)}(x). \quad (26)$$

2.5.4. Reproducibility

The implementation uses shared parameters across branches, fixed recursion depth D , and identical optimization settings across all experiments. No external pretraining or ensemble methods are used, and results are fully reproducible under fixed random seeds.

3. Theoretical Analysis

In this section we perform the Theoretical Analysis.

3.1. Computational Complexity Analysis

In this subsection, we analyze the forward-pass computational complexity of the proposed FANN architecture and compare it with a standard fully connected neural network.

Let

- n denote the input dimension;
- m denote the hidden layer width;
- D denote the recursion depth;
- b denote the branching factor.

For a conventional fully connected neural network with D hidden layers of width m , the dominant computational cost arises from matrix–vector multiplications. The forward-pass complexity is therefore

$$\mathcal{C}_{\text{ANN}} = \mathcal{O}(nm + Dm^2). \quad (27)$$

Since m^2 dominates for deep architectures, this becomes

$$\mathcal{C}_{\text{ANN}} = \mathcal{O}(Dm^2). \quad (28)$$

For the proposed FANN architecture, each recursion level generates b recursive branches. At recursion depth d , the effective connectivity density scales according to

$$\rho_d = b^{-d/\alpha}, \quad (29)$$

where α is the effective fractal dimension.

Therefore, the computational cost at depth d becomes

$$C_d = \mathcal{O}(m^2 \rho_d) = \mathcal{O}(m^2 b^{-d/\alpha}). \quad (30)$$

Summing over all recursion levels yields the total forward-pass complexity:

$$C_{\text{FANN}} = \mathcal{O}\left(nm + m^2 \sum_{d=0}^D b^{-d/\alpha}\right). \quad (31)$$

Using the geometric-series formula, we obtain

$$C_{\text{FANN}} = \mathcal{O}\left(nm + m^2 \frac{1 - b^{-(D+1)/\alpha}}{1 - b^{-1/\alpha}}\right). \quad (32)$$

Consequently, the proposed FANN architecture exhibits reduced computational growth compared with standard dense neural networks due to recursive sparsity and weight sharing.

In particular

- Larger values of α increase connectivity density and computational cost;
- Smaller values of α produce sparser recursive structures with lower computational complexity.

Therefore, FANN achieves multiscale recursive feature extraction while maintaining improved computational efficiency relative to conventional fully connected architectures of comparable depth.

3.2. Gradient Flow Analysis

The recursive structure of the proposed FANN architecture introduces shared transformations across multiple recursion levels. Consequently, the gradient propagation mechanism differs from that of conventional feedforward neural networks.

Let the network output at recursion depth d be denoted by

$$h^{(d)}(x) = \mathcal{F}^{(d)}(x), \quad (33)$$

and let $\mathcal{L}$ denote the training loss function.

Since the recursive branches share common parameters W , the total gradient with respect to W accumulates contributions from all recursion levels:

$$\frac{\partial \mathcal{L}}{\partial W} = \sum_{d=0}^D \frac{\partial \mathcal{L}}{\partial h^{(d)}} \frac{\partial h^{(d)}}{\partial W}. \quad (34)$$

Equation (34) shows that gradient information propagates simultaneously through multiple recursive scales. This multiscale accumulation improves information flow across the computational graph and reduces the sensitivity to isolated deep paths.

Using the recursive scaling coefficient $s \in (0, 1)$, the magnitude of the propagated gradient at depth d satisfies

$$\left\| \frac{\partial h^{(d)}}{\partial W} \right\| \leq s^d \|W\|^d. \quad (35)$$

Therefore, the total gradient norm can be bounded by

$$\left\| \frac{\partial \mathcal{L}}{\partial W} \right\| \leq \sum_{d=0}^D s^d \|W\|^d. \quad (36)$$

If

$$s\|W\| < 1, \quad (37)$$

the geometric series converges, yielding stable gradient propagation through recursive layers.

Hence, the recursive scaling mechanism acts as a natural regularization factor that mitigates exploding gradients while preserving multiscale information propagation. In addition, the shared recursive operators improve parameter efficiency by reusing transformations across different recursion depths.

3.3. Lipschitz Stability

In this subsection, we analyze the stability of the proposed FANN architecture using Lipschitz continuity. Stability is important to ensure that recursive feature propagation does not produce exponentially growing or vanishing activations.

Let $F : \mathbb{R}^n \rightarrow \mathbb{R}^m$ denote the recursive network operator. The network is said to be Lipschitz stable if there exists a constant $L_F > 0$ such that

$$\|F(x) - F(y)\| \leq L_F \|x - y\|, \quad \forall x, y \in \mathbb{R}^n. \quad (38)$$

Assume that the activation function σ is Lipschitz continuous with constant L_σ , i.e.,

$$\|\sigma(u) - \sigma(v)\| \leq L_\sigma \|u - v\|. \quad (39)$$

Using the recursive definition of the FANN architecture,

$$\mathcal{F}^{(d+1)}(x) = \sum_{k=1}^b s \mathcal{F}_k^{(d)}(x), \quad (40)$$

the Lipschitz constant at recursion depth d satisfies

$$L_d \leq s^d L_\sigma \|W\|^d. \quad (41)$$

Summing over all recursive scales yields the global stability bound

$$L_F \leq \sum_{d=0}^D s^d L_\sigma \|W\|^d. \quad (42)$$

If

$$s\|W\| < 1, \quad (43)$$

then Equation (42) becomes a convergent geometric series:

$$L_F \leq L_\sigma \frac{1 - (s\|W\|)^{D+1}}{1 - s\|W\|}. \quad (44)$$

Therefore, the recursive scaling coefficient $s \in (0, 1)$ acts as a stabilizing mechanism that controls the propagation of activations across recursive layers. Consequently, the proposed FANN architecture avoids uncontrolled amplification of signals while preserving multiscale recursive information flow.

3.4. Universal Approximation Property

The proposed FANN architecture is composed of affine transformations, recursive aggregations, and nonlinear activation functions. Consequently, it inherits the approximation capabilities of standard neural network architectures while introducing additional multiscale recursive representations.

Let $K \subset \mathbb{R}^n$ be a compact set and let

$$f : K \rightarrow \mathbb{R}$$

be a continuous function.

Since each recursive module of the FANN architecture consists of compositions of affine mappings and nonlinear activations, the recursive operator

$$\mathcal{F}^{(d)}(x)$$

belongs to the class of continuous nonlinear function approximators.

Furthermore, the multiscale aggregation

$$H(x) = \sum_{d=0}^D \omega_d h^{(d)}(x) \quad (45)$$

preserves continuity and enriches the representational capacity of the network across recursive scales.

Therefore, under sufficient width m and recursion depth D , the proposed FANN architecture can approximate arbitrary continuous functions on compact domains with arbitrary accuracy.

More precisely, for every continuous function $f \in C(K)$ and every $\varepsilon > 0$, there exists a FANN model $H(x)$ such that

$$\sup_{x \in K} |f(x) - H(x)| < \varepsilon. \quad (46)$$

Hence, the proposed FANN architecture satisfies a universal approximation property analogous to classical feedforward neural networks while additionally incorporating recursive multiscale feature representations and fractal connectivity structures.

3.5. Convergence Analysis

In this subsection, we discuss the convergence behavior of gradient-based optimization for the proposed FANN architecture.

Let $\mathcal{L}(W)$ denote the training loss function associated with the recursive network parameters W . Assume that

- The activation function σ is Lipschitz continuous;
- The loss function $\mathcal{L}$ is continuously differentiable;
- The recursive scaling coefficient satisfies $s \in (0, 1)$.

Using the recursive structure of the network, the parameter update generated by gradient descent is

$$W^{(t+1)} = W^{(t)} - \eta \frac{\partial \mathcal{L}}{\partial W^{(t)}}, \quad (47)$$

where $\eta > 0$ denotes the learning rate.

From the gradient flow relation,

$$\frac{\partial \mathcal{L}}{\partial W} = \sum_{d=0}^D \frac{\partial \mathcal{L}}{\partial h^{(d)}} \frac{\partial h^{(d)}}{\partial W}, \quad (48)$$

the recursive scaling factor introduces the bound

$$\left\| \frac{\partial \mathcal{L}}{\partial W} \right\| \leq \sum_{d=0}^D s^d C_d, \quad (49)$$

where C_d depends on the activation and weight norms.

Since $s < 1$, the recursive contributions form a bounded geometric series. Therefore, the gradient norm remains controlled during optimization:

$$\left\| \frac{\partial \mathcal{L}}{\partial W} \right\| < \infty. \quad (50)$$

Consequently, for sufficiently small learning rate η , the gradient descent iterations remain stable and converge locally toward a stationary point of the loss function.

Moreover, the recursive scaling mechanism smooths the effective optimization landscape by suppressing excessively large recursive contributions at deeper scales. This improves optimization stability and reduces the risk of exploding gradients in deep recursive architectures.

Therefore, the proposed FANN framework admits stable gradient-based optimization under standard regularity assumptions on the activation function and learning dynamics.

4. Numerical Experiments

In this section, we present numerical experiments conducted to evaluate the performance, parameter efficiency, and learning behavior of the proposed FANN.

4.1. Experimental Setup and Validation

This subsection presents the experimental configuration and validation protocol used to evaluate the proposed FANN. The experiments were designed to provide a fair, reproducible, and controlled comparison between the proposed architecture, a conventional convolutional neural network (CNN), and the fractal-inspired FractalNet architecture under identical training conditions.

4.2. Benchmark Dataset

The experiments were conducted on the CIFAR-10 benchmark dataset, which consists of 60,000 color images of size 32×32 distributed across 10 object classes. The dataset contains 50,000 training images and 10,000 testing images.

CIFAR-10 is widely used for evaluating deep convolutional architectures due to its moderate complexity and diverse visual categories.

4.3. Data Preprocessing

To ensure identical training conditions across all architectures, the same preprocessing and augmentation pipeline was applied to all models. The preprocessing procedure included:

- Channel-wise normalization using CIFAR-10 statistics;
- Random cropping with padding of 4 pixels;
- Random horizontal flipping with probability 0.5.

No augmentation was applied during evaluation.

4.4. Compared Architectures

Three neural architectures were evaluated:

- **ANN:** A conventional convolutional neural network baseline;
- **FractalNet:** A recursive fractal architecture based on self-similar branching;
- **FANN:** The proposed Fractal Artificial Neural Network incorporating recursive multi-scale operators and adaptive feature fusion.

The proposed FANN architecture introduces recursive fractal connectivity together with adaptive feature aggregation, aiming to improve parameter efficiency while preserving strong representational capability.

4.5. Training Configuration

All models were trained using identical optimization settings to ensure a fair comparison. The following hyperparameters were used throughout the experiments:

- Optimizer: Stochastic Gradient Descent (SGD) with momentum;
- Momentum: 0.9;
- Weight decay: 5×10^{-4} ;
- Initial learning rate: 0.1;
- Learning rate scheduler: cosine annealing;
- Batch size: 128;
- Number of epochs: 100;
- Loss function: Cross-Entropy Loss;
- Activation function: ReLU.

To stabilize training in the presence of sparse recursive connectivity, the weights of the proposed FANN architecture were initialized using a modified He initialization strategy adapted to the effective fractal connectivity density. In particular, the variance scaling factor was adjusted according to the active recursive fan-in at each fractal level, preventing gradient amplification or attenuation across recursive branches. This initialization improves optimization stability and accelerates convergence in deep fractal structures. All models were implemented using the PyTorch deep learning framework and trained on a CUDA-enabled NVIDIA Tesla T4 GPU.

4.6. Evaluation Metrics

Model performance was evaluated using the following criteria:

- Top-1 classification accuracy;
- Classification error rate;
- Number of trainable parameters.

The best testing accuracy obtained during training was reported for each model.

4.7. Experimental Results

The obtained experimental results are summarized in Table 1.

Table 1. Comparison of ANN, FractalNet, and the proposed FANN architecture on CIFAR-10.

Model	Parameters	Best Accuracy (%)	Error (%)
ANN	4,568,842	85.13	14.87
FractalNet	639,242	84.79	15.21
FANN	754,826	85.50	14.50

The proposed FANN architecture achieves competitive classification accuracy among the evaluated models while using substantially fewer trainable parameters than the conventional ANN baseline.

In particular, the ANN baseline requires approximately 4.57 million parameters, whereas the proposed FANN model achieves comparable or improved performance using only 0.75 million parameters. This indicates improved parameter efficiency arising from the recursive fractal connectivity and parameter-sharing mechanism.

Furthermore, the experimental results suggest that the adaptive fractal aggregation strategy contributes to stable optimization behavior and effective multiscale feature representation, compared with conventional fractal-inspired architectures.

4.8. Reproducibility

All experiments were conducted under consistent computational settings, ensuring identical preprocessing procedures, optimization configurations, and evaluation protocols across all compared models. No external pretraining, ensemble strategies, or test-time augmentation techniques were employed. The proposed methods were implemented entirely in Python using the PyTorch deep learning framework, enabling full transparency and reproducibility. All models were evaluated on the CIFAR-10 dataset under strictly identical training configurations, including optimizer type, learning rate schedule, batch size, and data preprocessing pipeline. The implementation, including model definitions and training scripts, is designed to allow exact replication of all reported results under the same experimental conditions.

4.9. Illustrative Regression Analysis

In addition to the large-scale CIFAR-10 classification experiments presented earlier, we further investigate the structural behavior and learning dynamics of the proposed FANN architecture using a representative nonlinear regression benchmark. Unlike the CIFAR-10 experiments, which provide the primary large-scale validation of the proposed model, this illustrative experiment is intended to analyze the influence of the fractal dimension parameter α , recursive connectivity, and parameter sharing in a controlled low-dimensional setting.

The objective of this analysis is to examine how recursive self-similar structures affect approximation accuracy, convergence behavior, and parameter efficiency. In the proposed FANN framework, the fractal dimension parameter α , controlled through the scaling factor s , governs the recursive branching topology and the density of hierarchical information propagation paths. By varying α , the complexity of recursive feature interactions and the degree of multiscale representation learning can be adjusted.

To study these effects, the proposed FANN architecture is compared with both a conventional feedforward ANN and the fractal-inspired FractalNet architecture under identical experimental settings. This analysis provides additional insight into the role of fractal topology in improving learning efficiency and representational capability while maintaining a compact parameterization.

4.10. Comparison with Standard ANN Architectures

To further evaluate the effectiveness of the proposed FANN architecture, we compare its performance with a conventional feedforward ANN using the same optimization settings and training conditions. The purpose of this experiment is to investigate the approximation capability, learning behavior, and parameter efficiency of recursive fractal architectures in a controlled regression setting.

A nonlinear synthetic regression dataset is generated according to

$$y = \sin(2\pi x) + 0.1\epsilon, \quad \epsilon \sim \mathcal{N}(0, 1), \quad x \in [0, 1]. \quad (51)$$

This benchmark represents a smooth nonlinear function corrupted by additive Gaussian noise and provides a suitable framework for analyzing approximation accuracy and convergence properties of neural models.

The baseline ANN consists of three fully connected hidden layers with ReLU activation functions. In contrast, the proposed FANN architecture employs recursive shared transformation blocks that generate self-similar hierarchical feature representations. The recursive structure is governed by the fractal dimension parameter α , which controls the effective complexity of the network topology. In the present study, three different fractal dimensions are considered:

$$\alpha = 1.94, \quad \alpha = 1.61, \quad \alpha = 1.36.$$

Unlike the conventional ANN, where each layer contains independent trainable weights, the proposed FANN shares parameters across recursive transformations. This recursive weight-sharing mechanism substantially reduces the total number of trainable parameters while preserving expressive representational capability. In this experiment, the ANN baseline contains 2209 trainable parameters, whereas each FANN configuration requires only 1153 parameters.

Figure 3 illustrates the regression outputs obtained from the ANN and the different FANN configurations. All models successfully approximate the underlying nonlinear structure of the target function. Despite using significantly fewer trainable parameters, the proposed FANN architectures generate smooth approximations that closely follow the sinusoidal behavior of the target data.

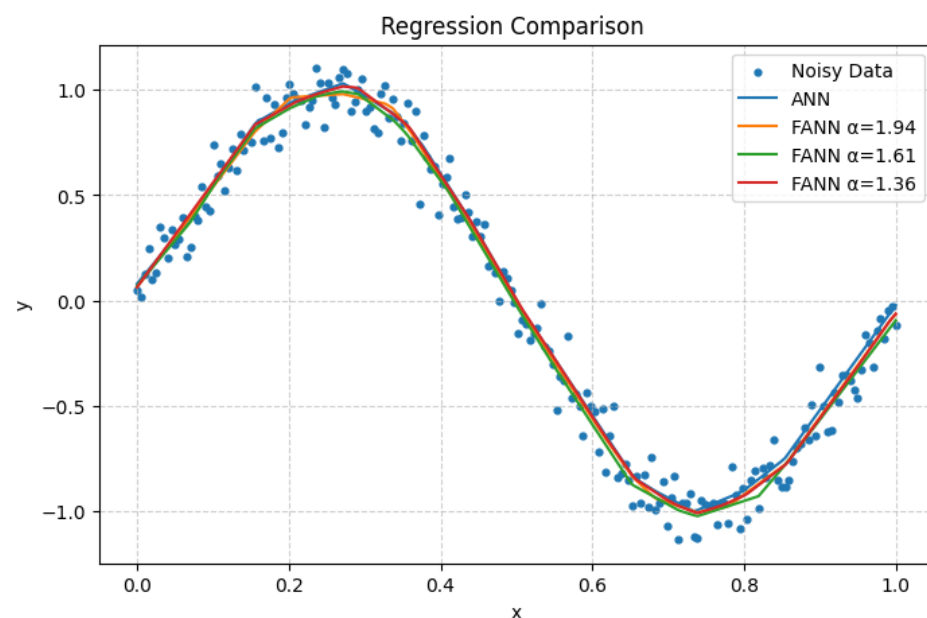


Figure 3. Regression outputs produced by the baseline ANN and the proposed FANN architectures for different fractal dimensions α . The proposed FANN models achieve accurate approximations of the nonlinear target function while using substantially fewer trainable parameters.

To further investigate the optimization behavior of the considered architectures, the mean squared error (MSE) loss was monitored throughout the training process over

300 epochs. The corresponding convergence curves are presented in Figure 4. The results demonstrate stable and consistent convergence for all models, indicating that both the conventional ANN and the proposed FANN architectures can be efficiently optimized using standard gradient-based training methods.

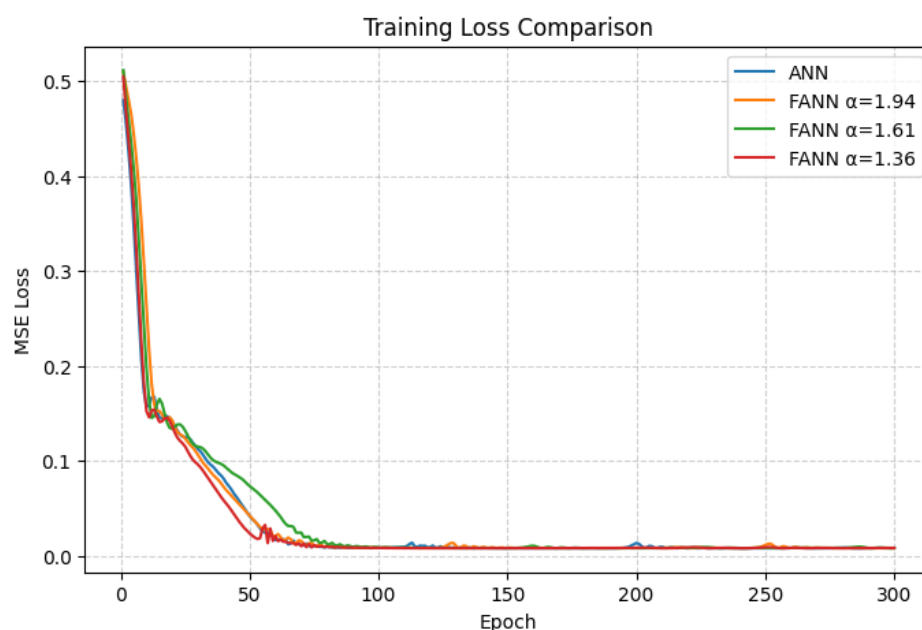


Figure 4. Training loss evolution of the baseline ANN and the proposed FANN architectures for different fractal dimensions during 300 optimization epochs.

Table 2 summarizes the quantitative performance metrics obtained for the ANN and FANN models. Among the investigated configurations, the FANN architecture with fractal dimension $\alpha = 1.61$ achieves the lowest final MSE while also reducing the overall training time compared with the baseline ANN.

These results indicate that the recursive self-similar structure of the proposed FANN provides improved approximation capability together with enhanced parameter efficiency. Despite using substantially fewer trainable parameters, the FANN models achieve comparable or better predictive performance than the conventional ANN architecture, demonstrating the effectiveness of fractal recursive representations for nonlinear function approximation.

Table 2. Performance comparison between the baseline ANN and the proposed FANN architectures.

Model	α	Parameters	Final MSE	Training Time (s)
ANN	—	2209	0.00817	0.6249
FANN	1.94	1153	0.00801	0.5478
FANN	1.61	1153	0.00797	0.4813
FANN	1.36	1153	0.00815	0.5231

The quantitative results presented in Table 2 demonstrate that the proposed FANN architecture achieves comparable and, in some cases, improved approximation accuracy relative to the conventional ANN while using nearly half the number of trainable parameters.

Among the investigated configurations, the FANN model with fractal dimension $\alpha = 1.61$ attains the lowest final MSE together with the shortest training time. These results indicate that the recursive self-similar structure of FANN provides an efficient representation mechanism for nonlinear function approximation.

The reduction in model complexity combined with competitive predictive performance suggests that fractal recursive connectivity can improve parameter efficiency while preserving stable learning behavior and accurate approximation capability.

4.11. Comparison with FractalNet Architectures

To further investigate the structural efficiency of the proposed framework, the FANN is compared with the fractal-inspired deep architecture FractalNet.

The comparison is performed using the same nonlinear regression problem defined in Equation (51). FractalNet constructs hierarchical representations through multiple parallel recursive paths, whereas the proposed FANN employs a recursive self-similar branching mechanism governed by the structural parameter α , which determines the effective fractal dimension and connectivity density of the network.

In contrast to conventional multi-path fractal architectures, the proposed FANN incorporates recursive feature reuse and adaptive hierarchical aggregation, enabling a more compact representation while preserving multiscale learning capability. This experiment is designed to analyze how the recursive fractal topology influences approximation accuracy, convergence behavior, and parameter efficiency relative to existing fractal-based neural architectures.

Figure 5 presents the regression approximations obtained using FractalNet and the proposed FANN architecture. Both models successfully capture the global nonlinear behavior of the target function. However, the proposed FANN produces more accurate local approximations and smoother predictive curves across different fractal dimensions. This improvement can be attributed to the recursive self-similar feature aggregation mechanism embedded within the FANN architecture, which enhances multiscale representation capability.

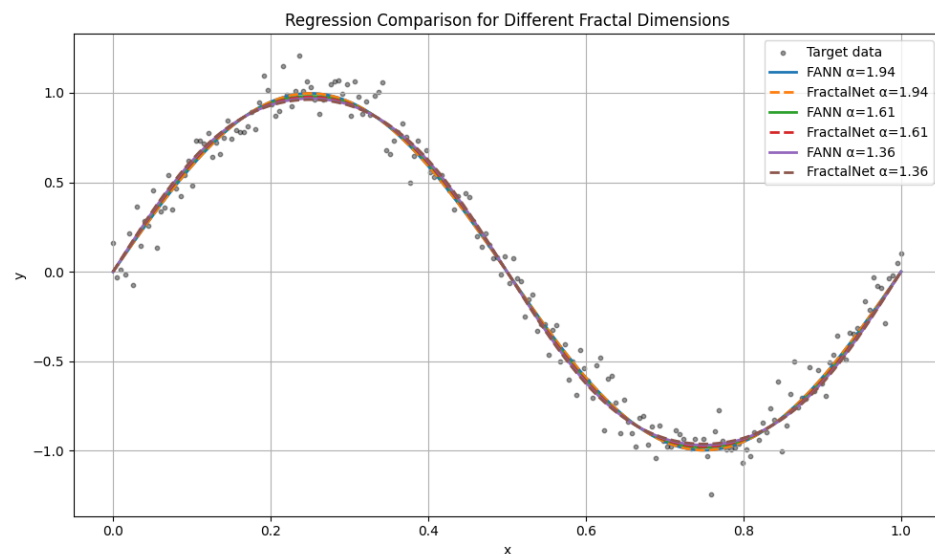


Figure 5. Regression approximations obtained using FractalNet and the proposed FANN architecture for different fractal dimensions on the nonlinear regression benchmark defined in Equation (51).

To further investigate the optimization behavior of both architectures, the mean squared error (MSE) loss was monitored during 300 training epochs. The corresponding training loss curves are shown in Figure 6.

As illustrated in Figure 6, both architectures exhibit stable convergence behavior throughout training. Nevertheless, the proposed FANN consistently converges to lower asymptotic loss values for all investigated fractal dimensions, indicating improved approximation efficiency and enhanced learning stability.

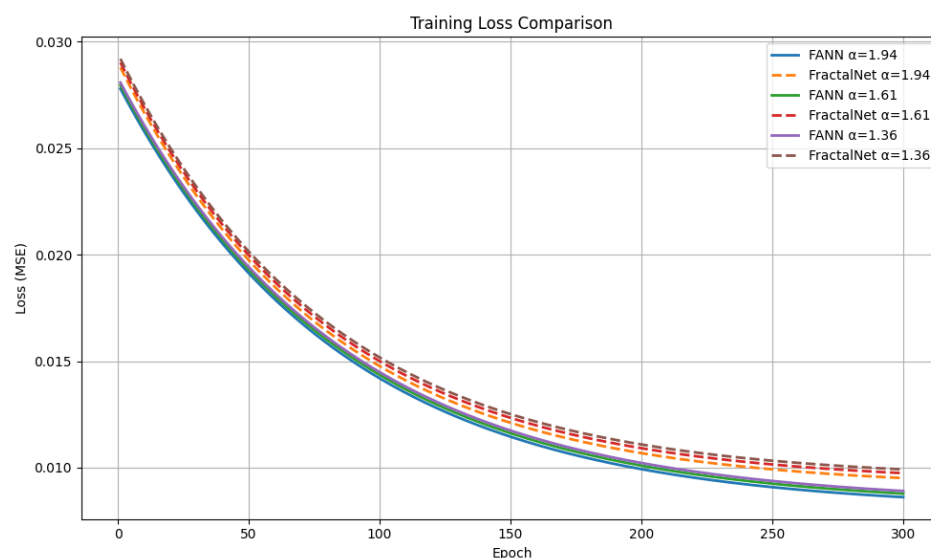


Figure 6. Training loss evolution for FractalNet and the proposed FANN architecture over 300 optimization epochs.

The quantitative performance results are summarized in Table 3. The proposed FANN architecture achieves lower final MSE values than FractalNet across all tested fractal dimensions while simultaneously requiring substantially fewer trainable parameters.

Table 3. Performance comparison between FANN and FractalNet for different fractal dimensions.

Model	α	Parameters	Final MSE	Training Time (s)
FANN	1.94	31	0.008425	1.01×10^{-5}
FractalNet	1.94	91	0.008432	1.78×10^{-5}
FANN	1.61	31	0.008747	2.09×10^{-5}
FractalNet	1.61	91	0.008863	1.16×10^{-5}
FANN	1.36	31	0.009171	9.29×10^{-6}
FractalNet	1.36	91	0.009449	9.17×10^{-6}

Overall, the experimental results demonstrate that the proposed FANN architecture provides a more compact and parameter-efficient representation while preserving strong approximation capability. The recursive self-similar topology governed by the fractal dimension parameter α enables efficient multiscale feature extraction without requiring a large number of trainable parameters. These findings highlight the potential of fractal structural design as an effective alternative framework for improving neural network efficiency and predictive performance.

5. Conclusions

In this work, we proposed the FANN, a recursive neural framework based on self-similar connectivity structures governed by the fractal dimension parameter α . The proposed architecture introduces hierarchical multiscale feature propagation while maintaining a compact parameterization through recursive weight sharing.

Experimental results on nonlinear regression tasks and the CIFAR-10 benchmark demonstrate that FANN achieves competitive performance compared with conventional ANNs and the fractal-inspired FractalNet architecture under comparable experimental settings. The results further indicate that the recursive structure can improve feature reuse and yield stable optimization behavior across different configurations.

We also analyzed the influence of the fractal dimension parameter α , showing that the network's connectivity topology has a significant impact on approximation accuracy and learning dynamics. These observations suggest that fractal-inspired recursive architectures provide a promising direction for designing structured and efficient representation learning models.

Future work will focus on extending the framework to larger-scale datasets and deeper recursive constructions, as well as developing a more rigorous theoretical analysis of fractal-based neural computation and its stability properties.

Author Contributions: Conceptualization, A.K.G.; methodology, A.K.G., C.S., P.E.T.J., and R.B.; investigation, A.K.G., C.S., P.E.T.J., and R.B.; software, A.K.G.; writing—original draft preparation, A.K.G.; writing—review and editing, C.S., P.E.T.J., and R.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The source code is available in a public GitHub repository: <https://github.com/alirezakhalili2005-design/FANN> (accessed on 5 June 2026).

Acknowledgments: During the preparation of this manuscript, the authors used ChatGPT 3.5 to assist with language editing and improving the clarity of the text. The authors reviewed and edited the generated content and take full responsibility for the final content of this publication.

Conflicts of Interest: The authors have no conflicts of interest to declare that are relevant to the content of this study.

Abbreviations

The following abbreviations are used in this manuscript:

FANN Fractal Architecture Neural Network
ANN Artificial Neural Network

References

1. Mandelbrot, B.B. *The Fractal Geometry of Nature*; WH Freeman: New York, NY, USA, 1982.
2. Falconer, K. *Fractal Geometry: Mathematical Foundations and Applications*; John Wiley & Sons: New York, NY, USA, 2004.
3. Jorgensen, P.E. *Analysis and probability: Wavelets, signals, fractals*; Springer Science & Business Media: New York, NY, USA, 2006; Volume 234.
4. Golmankhaneh, A.K. *Fractal Calculus and its Applications*; World Scientific: Singapore, 2022.
5. Haykin, S. *Neural networks and learning machines, 3/E*; Pearson Education India: Noida, India, 2009.
6. Yegnanarayana, B. *Artificial Neural Networks*; PHI Learning Pvt. Ltd.: New Delhi, India, 2009.
7. Hassoun, M.H. *Fundamentals of Artificial Neural Networks*; MIT Press: Cambridge, MA, USA, 1995.
8. Cartwright, H. *Artificial Neural Networks*; Springer: Cham, Switzerland, 2015; Volume 1260.
9. Braspenning, P.J. *Artificial Neural Networks: An Introduction to ANN Theory and Practice*; Springer Science & Business Media: Heidelberg, Germany, 1995; Volume 931.
10. Khanna, T. *Foundations of Neural Networks*; Addison-Wesley Longman Publishing Co., Inc.: Boston, MA, USA, 1990.
11. Mehrotra, K.; Mohan, C.K.; Ranka, S. *Elements of Artificial Neural Networks*; MIT Press: Cambridge, MA, USA, 1997.
12. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
13. Nielsen, M.A. *Neural Networks and Deep Learning*; Determination Press: San Francisco, CA, USA, 2015; Volume 25.
14. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444.
15. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
16. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
17. Larsson, G.; Maire, M.; Shakhnarovich, G. Fractalnet: Ultra-deep neural networks without residuals. *arXiv* **2016**, arXiv:1605.07648.
18. Liao, R.; Schwing, A.; Zemel, R.; Urtasun, R. Learning deep parsimonious representations. *Adv. Neural Inf. Process. Syst.* **2016**, *29*.

19. Chen, R.T.; Rubanova, Y.; Bettencourt, J.; Duvenaud, D.K. Neural ordinary differential equations. *Adv. Neural Inf. Process. Syst.* **2018**, *31*.
20. Sporns, O. Small-world connectivity, motif composition, and complexity of fractal neuronal connections. *Biosystems* **2006**, *85*, 55–64.
21. Singh, S.S.; Haobijam, D.; Malik, M.Z.; Ishrat, R.; Singh, R.B. Fractal rules in brain networks: Signatures of self-organization. *J. Theor. Biol.* **2018**, *437*, 58–66.
22. Jelinek, H.F.; Fernandez, E. Neurons and fractals: How reliable and useful are calculations of fractal dimensions? *J. Neurosci. Methods* **1998**, *81*, 9–18.
23. Gallos, L.K.; Sigman, M.; Makse, H.A. The conundrum of functional brain networks: Small-world efficiency or fractal modularity. *Front. Physiol.* **2012**, *3*, 123.
24. Di Lillo, S.; Marinucci, D.; Salvi, M.; Vigogna, S. Fractal and Regular Geometry of Deep Neural Networks. *arXiv* **2025**, arXiv:2504.06250.

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.