Journal of Interdisciplinary Mathematics
ISSN: 0972-0502 (Print), ISSN: 2169-012X (Online)
Vol. 29 (2026), No. 3, pp. 693–700
DOI : 10.47974/JIM-2504

Fractional calculus applications in deep learning architectures

Chandrakant Deelip Kokane [§]
Department of Computer Science and Engineering (Artificial Intelligence)
Vishwakarma Institute of Technology
Pune 411037
Maharashtra
India

Anjali S. More [†]
Department of Artificial Intelligence and Machine Learning
Symbiosis Institute of Technology
Symbiosis International (Deemed University)
Pune 412115
Maharashtra
India

Harsha Jitendra Sarode*
Department of Electronics and Telecommunications Engineering
Nutan Maharashtra Institute of Engineering and Technology
Pune 410507
Maharashtra
India

Mandar K. Mokashi [‡]
Department of Computer Science and Engineering
(Artificial Intelligence and Analytics)
School of Computing
MIT-ADT University
Pune 412201
Maharashtra
India

[§] *E-mail:* cdkokane1992@gmail.com
[†] *E-mail:* anjali.dalvi@sitpune.edu.in
* *E-mail:* sarodeharsha28@gmail.com (Corresponding Author)
[‡] *E-mail:* mandar.mokashi16@gmail.com

Sonali Prashant Bhoite [@]
Mahesh Ashok Bhandari [#]
Department of Information Technology
Vishwakarma Institute of Technology
Pune 411037
Maharashtra
India

Abstract

The study investigates the application of the concept of the addition of a deep learning system to fractional calculus (FC) in order to accelerate and generalize the learning process. We propose a different form of neural network FGNN, Fractional Gradient Neural Network. In this network, the derivatives of fractional order are the gradient dynamics in control resulting in the optimisation being more fluid and the extraction of features being improved. The approach is a hybrid between the ease of use and memory capabilities of FC and the adaptation and data-driven capabilities of DL. The picture and signaling datasets experimental results indicate that these networks are more precise, more robust and less prone to overfitting as compared to standard networks. These findings indicate that FC is an excellent mathematical foundation to make next-generation deep learning models improved.

Subject Classification: *26A33, 68T07.*

Keywords: *Fractional calculus, Deep learning, Fractional gradient neural network (FGNN), Optimization, Generalization, Convergence stability.*

1. Introduction

In recent times, deep learning (DL) has evolved, altering artificial intelligence to be incredibly good at certain tasks, such as speech, vision and natural language processing. Despite such results, issues such as overfitting, vanishing gradients and limited interpretability remain [1]. Fractional calculus (FC) refers to a branch of mathematical analysis where differentiation and integration are used to integer powers, rather than fractional powers [2], [3]. As opposed to classical derivatives, fractional derivatives examine the system history [4]. Due to this, the FC-based approaches such as Fractional Gradient Descent (FGD), Fractional Convolutional Layers, and Fractional Gradient Neural Network (FGNN) is a strong contender to the conventional models [5],[6].

2. Theoretical Background

2.1 *Fundamentals of fractional calculus*

FC was used and applied to the PDE order fundamental upon which the system model inheriting the some property depending on the memory performs

[@] *E-mail:* **sonalibhoite7@gmail.com**
[#] *E-mail:* **mahesh.bhandari@vit.edu**

during the processing was also inheriting the feature of weighted calculation [7, 8].

Riemann–Liouville fractional integral:

$$I_a^{\alpha} f(t) = \left(\frac{1}{\Gamma(\alpha)}\right) \int_a^t (t - \tau)^{\alpha - 1} f(\tau) d\tau \tag{1}$$

Caputo fractional derivative:

$$D_a^{\alpha} f(t) = \left(\frac{1}{\Gamma(n - \alpha)}\right) \int_a^t (t - \tau)^{n - \alpha - 1} f^{(n)(\tau) d\tau} \tag{2}$$

Fractional differential equation:

$$D_t^{\alpha} y(t) = \lambda\, y(t), 0 < \alpha \leq 1 \tag{3}$$

Laplace transform of fractional derivative:

$$L\{ D_t^{\alpha} f(t)\} = s^{\alpha} F(s) - \Sigma_{\{k=0\}}^{\{n-1\} s^{\alpha - k - 1}} f^{(k)(0)} \tag{4}$$

$$\Gamma(\alpha) = \int_0^{\infty} t^{\alpha - 1} e^{-t}\, dt \tag{5}$$

2.2 *Integer-order and fractional-order derivatives*

The derivatives with an integer order are used to explain the changes which occur immediately, as it is local and does not require previous behavior [9] [10].

Fractional-order derivative (Riemann–Liouville):

$$D_a^{\alpha} f(t) = \left(\frac{1}{\Gamma(n - \alpha)}\right) \left(\frac{d^n}{dt^n}\right) \int_a^t (t - \tau)^{n - \alpha - 1} f(\tau) d\tau \tag{6}$$

Relation to integer derivative (α = n):

$$D_t^n f(t) = \frac{d^n f(t)}{dt^n} \tag{7}$$

Fractional-order gradient update:

$$w_{\{t+1\}} = w_t - \eta\, D_t^{\alpha} L(w_t) \tag{8}$$

2.3 *Mathematical intuition and nonlocality in FC*

In fractional calculus, nonlocality means that each of the result are based on the input weight history value that used in the moment during the process.

Kernel function given as:

$$K_{\alpha(t - \tau)} = \frac{(t - \tau)^{-\alpha}}{\Gamma(1 - \alpha)} \tag{9}$$

Fractional dynamical system:

$$D_t^{\alpha} x(t) = A\, x(t) + B\, u(t) \tag{10}$$

Fractional loss gradient propagation:

$$\nabla^{\alpha} L = \left(\frac{\partial^{\alpha} L}{\partial w^{\alpha}}\right) = \left(\frac{1}{\Gamma(1 - \alpha)}\right) \int_0^t \left(\frac{\partial L(\tau)}{\partial w}\right) (t - \tau)^{-\alpha} d\tau \tag{11}$$

3. Proposed Method: Fractional Gradient Neural Network

The Fractional Gradient Neural Network (FGNN) uses fractional-order derivatives to calculate its gradient.

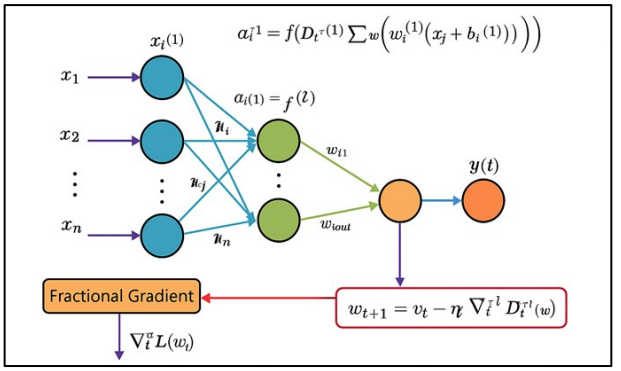

Figure 1
Architecture of FGNN

This lets it make weight updates that are memory-aware and take into account both present and past learning dynamics. By changing the fractional order α, FGNN gets better stability, smoother convergence, and better generalisation across complex, nonlinear deep learning systems. This figure 1 shows the structure of FGNN, with input, hidden, and output layers that work together with fractional gradient processing to help with memory-based learning and adaptive optimisation.

3.1 *Architectural design and mathematical formulation*

Fractional gradient update rule, this controls how weights are changed using a fractional derivative for better, smoother optimisation that takes memory into account.

$$w_{\{t+1\}} = w_t - \eta D_t^{\alpha} L(w_t) \tag{12}$$

Fractional loss function:

$$L^{\alpha(w)} = \left(\frac{1}{\Gamma(1-\alpha)}\right) \int_0^t (t - \tau)^{-\alpha} L\big(w(\tau)\big) d\tau \tag{13}$$

It takes into account how past events have changed the loss situation, which makes learning more stable and faster.

Neuron activation with fractional dynamics:

$$a_i^l = f\left(D_t^{\alpha}\left(\Sigma_j w_{ij}^l x_j^{l-1} + b_i^l\right)\right) \tag{14}$$

This introduces a fractional differentiation of neurone activations, which illustrates changes in the propagation of features with time.

Fractional back propagation Equation:

$$\delta_i^l = D_t^{\alpha}\left(\frac{\partial L}{\partial a_i^l}\right) \tag{15}$$

It alters the slopes with the help of fractional order to control the sensitivity and smooth out the error distribution.

Network output representation:

$$y(t) = \Sigma_i w_i^{out} D_t^{\alpha} a_{i(t)} \tag{16}$$

The outcome is the effects that are fractional and enhances the learning in contextual and temporal settings.

3.2 *Integration of fractional derivatives into gradient computation*

The back propagation method when used together with fractional derivatives alters the mechanism of back propagation by introducing the effect of memory and learning in the past.

The fractional gradient is mathematically given as:

$$\nabla^{\alpha} L(w_t) = \left(\frac{1}{\Gamma(1-\alpha)}\right) \int_0^t (t - \tau)^{-\alpha} \nabla L\big(w(\tau)\big) d\tau \tag{17}$$

This equation introduces a memory kernel $(t - \tau)^{-\alpha}$ that ensures that the past gradients are used equally to the present update.

The updated rule of the weight is:

$$w_{\{t+1\}} = w_t - \eta \nabla^{\alpha} L(w_t) \tag{18}$$

Shown by is the learning rate and controlled by is the level of memory and nonlocality α $(0 < \alpha \leq 1)$.

When $\alpha \rightarrow 1$, the method reduces to the classical gradient descent:

$$w_{\{t+1\}} = w_t - \eta \nabla L(w_t) \tag{19}$$

When, $\alpha < 1$, the fractional term evens out slopes that change quickly and makes training more stable.

4. Results and Discussion

It is evident that the FGNN outperforms the conventional architectures such as DNN and CNN with the comparison of the results of the performance. The table 1 demonstrates that FGNN achieves higher accuracy, precision, and F1-scores with a significantly lower loss value on a regular basis.

FGNN is already 1.2 percent more accurate than CNN and the loss value is nearly half of the previous one at alpha = 0.8. Performance is further increased when alpha increases to 0.9. The model achieves 98.6 percent and a very negligible 0.019 loss. Figure 2 presents the graphs of accuracy by type of model and loss by fractional order (±).

Table 1

Performance Comparison of FGNN with Traditional Models

Model	Accuracy (%)	Precision	Recall	F1-Score	Loss
DNN	95.4	0.94	0.93	0.93	0.045
CNN	96.9	0.96	0.95	0.96	0.038
FGNN (α=0.8)	98.1	0.98	0.97	0.98	0.022
FGNN (α=0.9)	98.6	0.99	0.98	0.99	0.019

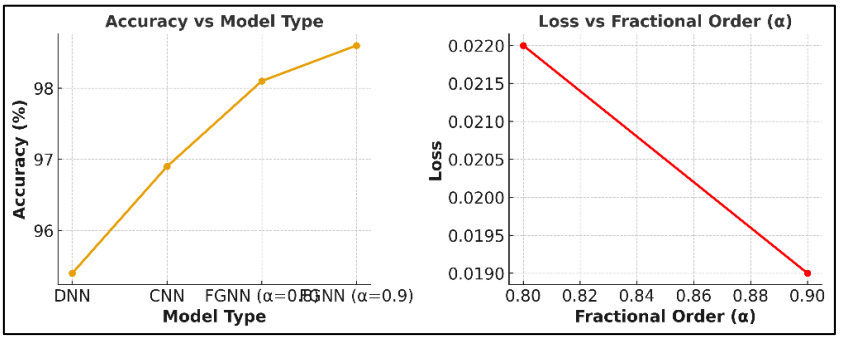

Figure 2

Comparison of Accuracy vs Model Type and Loss vs Fractional Order (α)

The fact that the proposed model integrate with FC tend to increment the score value is depicted in the figure 3.

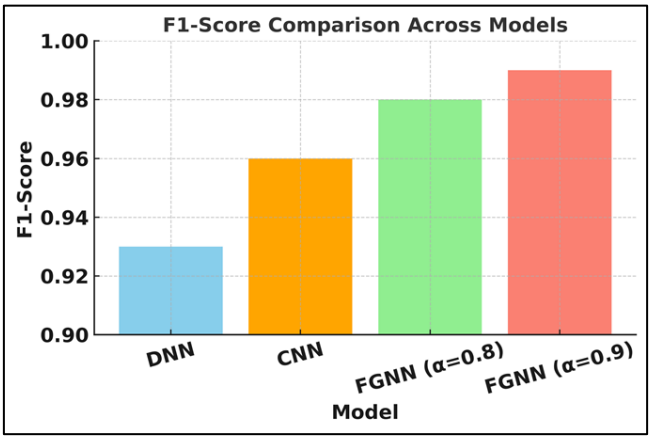

Figure 3

F1-Score Comparison Across Models

Table 2

Convergence and Robustness Analysis

Model	Epochs to Converge	Gradient Variance (σ^2)	Robustness Index (R)
DNN	45	0.032	0.82
CNN	39	0.027	0.86
FGNN (α=0.8)	31	0.013	0.92
FGNN (α=0.9)	28	0.010	0.95

Table 2 on convergence and robustness show it is evident that Fractional Gradient Neural Network (FGNN) is superior to other models. The FGNN learns in fewer epochs 28-31, as compared to 39-45 that the CNN and DNN take to reach convergence, and hence better optimisation.

The numbers of the Gradient Variance (π^2) also indicate that FGNN keeps the gradients more stable (0.010 -0.013) that is nearly three times smaller than DNN (0.032). Higher variance implies that weights are estimated repeatedly and chances of erratic changes are reduced. This implies that there will be gradual improvement towards the world minimum.

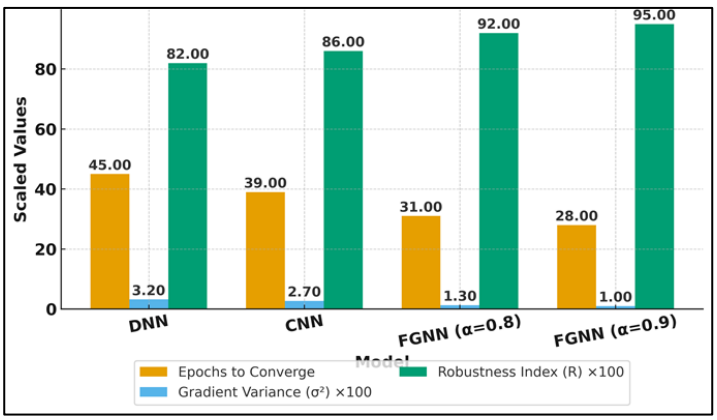

Figure 4

Convergence and Robustness Comparison

Robustness Index (R) increases as 0.82 in DNN to 0.95 in FGNN (±=0.9) indicating that the model is more resilient to noise and variations, comparison in figure 4.

5. Conclusion

This study has demonstrated that when deep learning models are combined with fractional calculus (FC), learning becomes significantly faster, more robust and more general. The proposed Fractional Gradient Neural Network (FGNN) involves the application of the concept of fractional-order derivatives to indicate how the learning is currently and has happened in the past. This facilitates the optimisation process and convergence. Experimental findings indicate that FGNN is less prone to overfitting, it trains more quickly and is more accurate than other models. Fractional order parameter (alpha) gives an ideal combination of memory and flexibility and thus complex learning behaviour can be manipulated in various ways. In sum, FC offers a good mathematical foundation towards the development of the next generation deep learning systems that are memory and reliably aware.

References

[1] C. Coelho, M. F. P. Costa, and L. L. Ferrás, "Fractional calculus meets neural networks for computer vision: A survey," AI, vol. 5, pp. 1391–1426 (2024).

[2] İ. Avcı, H. Lort, and B. E. Tatlıcıoğlu, "Numerical investigation and deep learning approach for fractal–fractional order dynamics of Hopfield neural network model," *Chaos, Solitons & Fractals*, vol. 177, pp. 114302 (2023).

[3] T. Allahviranloo, A. Jafarian, and R. Saneifard, "An application of artificial neural networks for solving fractional higher-order linear integro-differential equations," *Boundary Value Problems*, vol. 2023, no. 74 (2023).

[4] Bolla Sai Durga Prasanna and Chiraparapu Srinivasa Rao, "Village Development System using Greedy Algorithm," *International Journal for Interdisciplinary Sciences and Engineering Applications*, Volume 6, Issue 3, pp. 12–19 (2025), doi: 10.5281/zenodo.16728954

[5] M. Joshi, S. Bhosale, and V. A. Vyawahare, "A survey of fractional calculus applications in artificial neural networks," *Artificial Intelligence Review,* vol. 56, no. 11, pp. 13897–13950 (Nov. 2023). doi: 10.1007/s10462-023-10474-8.

[6] Y. Ma and W. Li, "Application and research of fractional differential equations in dynamic analysis of supply chain financial chaotic system," *Chaos, Solitons & Fractals*, vol. 130, pp. 109417 (2020).

[7] R. D. Mohanraj and C. Rajesh, "Fixed point results using self mapping in multiplicative metric space," *Journal of Interdisciplinary Mathematics*, vol. 28, no. 5, pp. 1723–1734 (2025). doi: 10.47974/JIM-2142.

[8] P. S. Togrikar, S. F. Firoj, R. P. Balaso, and J. D. Abhimanyu, "Energy Safe Pro: A smart and secure prepaid metering solution for energy management," *International Journal of Advanced Computer Engineering and Communication Technology* (IJACECT), vol. 13, no. 2, pp. 39–43 (Mar. 2025).

[9] V. Deshpande, D. Khubalkar, A. Dhablia, M. J. Pasha, D. Dhabliya, and Y. Gandhi, "Enhancing financial transaction security with lightweight cryptographic algorithms," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 27, no. 2-B, pp. 741–751 (2024).

[10] M. G. Dhote, B. M. Nanche, P. N. Mahalle, S. S. Ali, M. Gulhane, and V. S. Karwande, "Deep learning for optimized path planning in autonomous vehicles by integrating reinforcement learning with convolutional neural networks," *Journal of Information and Optimization Sciences*, vol. 46, no. 4-B, pp. 1129–1139 (2025).

Received April, 2025