AIMS Mathematics, 10(11): 26717–26743.
DOI: 10.3934/math.20251175
Received: 21 September 2025
Revised: 16 October 2025
Accepted: 27 October 2025
Published: 18 November 2025

https://www.aimspress.com/journal/Math

Research article

Predictive modeling of complex networks using deep learning and fractional dynamics

Muhammad Ayaz[1,2], **Tariq Ali**[1,2,*], **Mohammad Hijji**[2], **Imran Baig**[3], **Tareq Alhmiedat**[1,2] **and El-Hadi M. Aggoune**[1]

[1] Artificial Intelligence and Sensing Technology Research Center (AIST), University of Tabuk, Tabuk 71491, Saudi Arabia
[2] Faculty of Computers and Information Technoloy, University of Tabuk, Tabuk 71491, Saudi Arabia
[3] Senior Lecturer in Computer Science, Cardiff School of Technologies, Cardiff Metropolitan University, Llandaff Campus, Western Ave, Cardiff CF5 2YB, United Kingdom

* **Correspondence:** Email: teshaq@ut.edu.sa.

Abstract: The increasing complexity of modern networks, from communication infrastructures to power grids and social networks, demands models that capture both structural dependencies and nonlinear dynamics of long memory. We proposed a hybrid framework that unified deep learning (graph neural networks, recurrent/attention modules) with fractional calculus to model nonlocal memory, anomalous diffusion, and self-similarity. Fractional differential formulations provide a principled description of network evolution, for which we stated a checkable stability condition; the learning pipeline coupled gradient-based training with fractional operators for robust, interpretable prediction. On Los Angeles metropolitan area traffic (METR-LA) dataset, the proposed ensemble integrated deep fractional model (EIDFM) achieved mean absolute error (MAE) around 6.4 and root mean square error (RMSE) 10.8, which showed improvement over the strongest baseline hybrid (CNN-LSTM): MAE 7.8, RMSE 12.5) by 18% and 14%, respectively; mean absolute percentage error (MAPE) droped from 6.3% to 5.2% ($\approx$ 17% relative), while R^2 rose from 0.91 to 0.94. Results were reported as mean±std over five seeds with paired significance tests. A lightweight efficiency analysis showed modest overhead relative to the baseline (inference 3.2 ± 0.3 *ms/step* vs. 2.8 ± 0.2; parameters 12.8M vs. 11.3M), justified by the accuracy gains. These findings indicated that integrating fractional operators with graph-based deep learners yielded a mathematically grounded and practically effective paradigm for understanding and managing complex network dynamics.

Keywords: complex networks; deep learning; fractional calculus; network dynamics; nonlinear systems; mathematical modeling
Mathematics Subject Classification: 34A08, 68T07

1. Introduction

Complex networked systems–such as communication infrastructures, transportation grids, power networks, and social interaction graphs–frequently exhibit nonlinear dynamics, with memory, nonlocal effects, and anomalous diffusion. Traditional integer-order differential (and difference) models often fail to capture such behaviors, especially when past states have long-lasting influences or when spatial/graph structure influences propagation over time. To model these phenomena more faithfully, there is growing interest in combining fractional calculus, which generalizes differentiation and integration to non-integer (fractional) orders, with machine learning/deep learning architectures capable of capturing complex patterns and high-dimensional structure.

Recent studies show promising advances along this direction. For example, Kang et al. [1] introduced the fractional-order graph neural dynamical network (FROND), which uses Caputo fractional derivatives in continuous graph neural networks (GNNs) to better model dynamics with memory effects over graphs, outperforming integer-order differential baselines. Additionally, in [2], the authors proposed a distributed-order fractional graph operating network (DRAGON), a graph neural network (GNN) framework that integrates distributed-order fractional calculus to flexibly model derivative order as a continuous distribution rather than a fixed scalar. There are also works on neural variable-order fractional differential equations (NvoFDE) that allow the fractional order to vary over time or state, thereby improving the expressiveness of systems with evolving memory demands [3].

Meanwhile, classical control systems and neural networks have embraced fractional and fractal tools in Hopfield networks with fractal-fractional derivatives for capturing more intricate state dependencies [4,5]. Another line uses data-driven frameworks to discover fractional differential equations in complex systems, estimating both the sparse structure and the fractional order from data [6,7].

1.1. Gaps and motivation

Despite these advances, there remain key limitations:

- Most existing models use fixed fractional order (or distributed, but not yet well integrated with deep learning) and don't fully combine fractional operators, fractal/long-memory measures, and graph structural learning.
- Scalability and training efficiency are often issues–many fractional models have high computational cost or require specialized solvers.
- Applications to real large-scale complex networks (with graph topology, structural heterogeneity, nonlocal interactions) are relatively few; there is a need for frameworks that are both mathematically rigorous and practical.

1.2. Contributions

The following are the key contributions of this study to bridge these gaps.

(1) We propose an ensemble integrated deep fractional model (EIDFM) combining (i) graph-structured deep learning (GNNs, recurrent/temporal models), (ii) fractional calculus (both fixed and variable order), and (iii) fractal/long-memory measures, to model complex network dynamics with memory and structural effects.

(2) We derive fractional differential equations governing network evolution, and provide theoretical analysis (existence, uniqueness, stability) for these equations under our model assumptions.
(3) We develop efficient learning and training procedures that integrate fractional operators into deep networks, including techniques to reduce computational overhead and ensure scalability for large networks.
(4) We conduct experiments on synthetic and real networked systems (e.g., traffic networks, diffusion on graphs) showing that our approach improves prediction accuracy, captures memory effects (via fractional orders), and yields interpretable insights into network dynamics compared to pure integer-order or standard machine learning models.
(5) We discuss the trade-offs between model complexity, fractional order selection, and interpretability, offering guidelines for when the fractional + deep learning combination is most beneficial.

1.3. Structure of the paper

The rest of the paper is organized as follows. In Section 2, we review the mathematical background: fractional derivatives, fractal measures, and graph neural networks. In Section 3, we present data preprocessing. In Section 4, we present our proposed model formulation and learning algorithms. In Section 5, we describe the proposed architecture. In Section 6, we present the experimental results. Finally, Section 7 concludes the paper and suggests future work.

2. Literature review

The study of complex network dynamics has increasingly turned to integrating deep learning and fractional calculus to capture memory effects, anomalous diffusion, and nonlocal interactions. Classical integer-order models, such as those based on ordinary differential equations, often fail to represent the long-memory dependencies observed in communication networks, traffic systems, and power grids [8]. To address this limitation, recent research has incorporated fractional-order derivatives into GNNs and related models. For example, Zhao et al. introduced the FROND, which embeds Caputo fractional derivatives into continuous GNNs to mitigate over-smoothing and better capture temporal memory in node feature propagation [9]. While FROND demonstrated significant improvements over standard GNNs, it relies on a fixed fractional order, which may not sufficiently adapt to the heterogeneity of real-world network dynamics.

Fractional calculus has been proven to be a valuable tool for adding sophistication to neural models that goes beyond classical integer-order counterparts. It is shown in [10] that extending activation functions with fractional concepts leads to improvements in representational ability and stability during training. As shown in [11] for the case of computer vision, fractional differential operators improve edge awareness, texture description, and regularization. According to survey results reported in [12–14], fractional-order neural networks can better model memory effects, long-range dependence, and anomalous dynamics. This leads to more accurate predictions and performance in control, forecasting, and signal-processing applications. At the level of interconnected systems, Li et al. [15] studies synchronization in fractional complex networks with unbounded coupling delays.

To address the limitations of fixed-order models, Hasani et al. [16] also introduced the DRAGON, a more generalized framework that learns a distribution over derivative orders rather than

a fixed value [17]. This approach offers greater flexibility for capturing diverse memory effects across nodes and edges and has been shown to outperform existing methods across a series of benchmark graph learning tasks [18]. However, it comes with an increased overhead of training and interpretability, since the learned distribution of derivative orders does not necessarily have a clear physical explanation [19–21].

In addition, Avcı studied the stability, chaos, and long memory of Hopfield neural networks with fractal-fractional derivatives from a more mathematical, proof-oriented perspective [22–24]. These works, however, have mainly focused on theoretical and mathematical proofs of stability and other dynamical properties and have been limited to small-scale networks without large-scale data-driven experiments. Another line of works by [25–27] focused on the use of fractional calculus in convolutional neural networks (CNNs) for computer vision tasks, including image enhancement [28–30], image denoising, and segmentation. The results in these works demonstrated the potential of fractional operators to capture spatial non-locality, but, to the best of our knowledge, have not yet been explored for graph-structured or dynamical networks [31–33].

A broader overview of fractional calculus in machine learning was provided by Raubitzek et al., who reviewed applications ranging from feature augmentation to physically informed models [5, 34]. Their survey concludes that while fractional derivatives have been integrated into machine learning pipelines, very few works explicitly address combining fractional differential equations with graph neural networks or complex network evolution. Furthermore, the majority of existing approaches face challenges related to computational cost, as fractional-order operators are expensive to compute, and scalability to large dynamic graphs remains a bottleneck. Additionally, interpretability issues persist, as the choice of fractional order or distribution is often heuristic and lacks systematic justification.

With recent data-driven learning for networked systems, the iteration of the study policy with closed-loop stability and performance guarantees for unknown nonlinear systems is presented in [35]. Data-driven control triggered by distributed events in networked systems provides stability certificates via linear matrix inequalities (LMIs) and looped functionals [36]. Complementary lines include Koopman embeddings for grid control [37] and online learning for switched systems with stability [38, 39]. These efforts align with our goal of theory-based network learning and motivate our fractional-deep hybridization. The proposed predictor is modular and can be paired with physics-informed operators from computational mechanics, for example, the extended finite element method (XFEM) and the boundary element method (BEM), by injecting boundary/interface constraints and operator-consistent graph filters; exploring such couplings is a promising direction for multiphysics networks [40].

Table 1 provides a critical comparison of existing state-of-the-art methods, revealing that while FROND offers substantial performance improvements with fixed fractional orders, it does not adapt to heterogeneous network structures. DRAGON extends this by adopting distributed-order derivatives, but at the expense of interpretability and training cost. Hopfield networks with fractal-fractional derivatives offer theoretical insights but lack practical scalability. Fractional CNNs in computer vision demonstrate the potential of fractional operators in learning tasks, yet they do not address graph-based or temporal network applications. These gaps motivate the development of hybrid frameworks that unify fractional calculus, fractal analysis, and deep learning architectures to provide both mathematical rigor and practical effectiveness in modeling complex network dynamics.

Table 1. Comparison of recent studies on fractional calculus, deep learning, and network dynamics.

Reference	Focus	Methods used	Limitations
Kang et al. (2024) [1]	Coupling GNNs with fractional-order dynamics	GNNs integrated with fractional differential equations	Limited to small-scale networks, lacks real-time validation
Sivalingam (2025) [7]	Neural fractional-order differential equations	Fractional differential equation solvers embedded in neural nets	Computationally expensive, stability not fully proven
Santos et al. (2025) [8]	Fractional neural operators	Symmetrized Caputo operators in neural architectures	Theoretical framework, lacks large experimental validation
Cui et al. (2025) [5]	Variable-order fractional networks	Variable-order, forward deployed engineering models with neural approximation	Complexity increases with order, scalability concerns
Joshi et al. (2023) [12]	Survey on fractional calculus in artificial neural networks	Literature review of fractional activation functions and optimization	Mainly survey, no new hybrid framework proposed
Liu et al. (2025) [13]	Synchronization in fractional complex networks	Fractional-order dynamic equations with coupling delays	Focused on synchronization, not forecasting or learning
Yang et al. (2025) [23]	Epidemic dynamics in networks	Fractional-order epidemic spreading models	Narrow application domain (epidemic only)
Li et al. (2023) [28]	Fractional Laplacian GNNs	Fractional Laplacian operator applied to GNNs	Focused on graph structure only, lacks temporal forecasting
Tiwari et al. (2023) [31]	Neural operators for fractional dynamics	Deep operator networks for fractional partial differential equations	High computational cost, requires large training datasets
Viera-Martin et al. (2022) [26]	Hybrid fractional-deep models	Combines fractional models with deep learning for spatiotemporal forecasting	Tested on limited datasets, ensemble integration missing
Baleanu et al. (2023) [27]	Fractional calculus in machine learning and control	Applied fractional derivatives in neural learning	Lacks large-scale empirical validation
Mou et al. (2019) [34]	Fractional differential equations in network science	Fractional-order dynamics in complex networks	No deep learning integration, mainly theoretical
Anghinoni et al. (2019) [33]	Deep learning for network forecasting	Convolutional neural network-long short-term memory (CNN-LSTM) applied to time-series in networks	Does not exploit fractional features
Waikhom et al. (2023) [30]	Survey on GNNs	Comprehensive review of GNNs	Does not consider fractional or fractal integration
Wu et al. (2021) [18]	GNN architectures	Temporal/dynamic GNNs for evolving networks	Limited interpretability, no fractional calculus
This work (2025)	Hybrid fractional-deep forecasting in complex networks	Fractional-order models + fractal features + deep learning (CNN, LSTM, GNN) + ensemble integration	Addresses scalability, interpretability, and robustness gaps identified in prior works

3. Problem setup, assumptions, and operators

System class. We consider a networked nonlinear discrete-time system on a graph $G = (V, E)$ with $n = |V|$ nodes:

$$x_{t+1} = \Phi(x_t, u_t, A) + w_t, \qquad y_t = Hx_t + \varepsilon_t, \tag{3.1}$$

where $x_t \in \mathbb{R}^{nd}$, $u_t \in \mathbb{R}^m$, A is a (row-)normalized adjacency or Laplacian-derived matrix, H is a measurement map, w_t is a bounded process disturbance, and ε_t is measurement noise.

Assumptions. (A1) $\Phi(\cdot)$ is globally Lipschitz in x with constant L_Φ.
(A2) $\|A\|_2 \le 1$ (normalized graph operator).
(A3) $\|w_t\|_2 \le \sigma_w$.
(A4) ε_t is zero-mean sub-Gaussian with variance proxy σ_ε^2.
(A5) Inputs are bounded $\|u_t\| \le U_{\max}$.
(A6) Train/val/test splits are i.i.d. across seeds.

Fractional/discrete operators. (i) Caputo fractional derivative ${}^C D_t^\alpha$ with order $\alpha \in (0, 1)$, discretized by the L1 scheme on a uniform grid Δt:

$$ {}^C D_t^\alpha x(t_k) \approx \frac{1}{\Gamma(2-\alpha)\,\Delta t^\alpha}\left[x_k - \sum_{j=0}^{k-1} b_j^{(\alpha)} x_{k-1-j}\right], \quad b_j^{(\alpha)} = (j+1)^{1-\alpha} - j^{1-\alpha}.$$

(ii) Distributed-order operator $\int_0^1 w(\alpha)\, {}^C D_t^\alpha x\, d\alpha$ via Q-point Gauss-Legendre quadrature $\sum_{q=1}^{Q} \omega_q\, {}^C D_t^{\alpha_q} x$;
(iii) Fractional differencing $(1-B)^d = \sum_{k=0}^{K} \binom{d}{k}(-1)^k B^k$ (truncated at lag K) in fractional autoregressive integrated moving average (FARIMA) and fractional exponential time series (FETS);
(iv) Graph fractional Laplacian L^β, $0 < \beta < 1$, implemented with Chebyshev/Krylov approximation.

Predictor class.

$$\hat{y}_t = \sum_{i=1}^{M} \omega_i \hat{y}_t^{(i)} + g\Big(f_t, h_t^{\mathrm{GNN}}, h_t^{\mathrm{CNN}}, h_t^{\mathrm{LSTM}}; \theta\Big),$$

where FARIMA/FETS, multifractal detrended fluctuation analysis (MFDFA) and Hurst exponent outputs comprise f_t, deep modules produce $h_\cdot$, $\omega_i \ge 0$, $\sum_i \omega_i = 1$, and $g(\cdot)$ is a Lipschitz meta-learner.

End-to-end error bound and stability. Let L_{DL} be a Lipschitz constant for the deep block, L_g for $g(\cdot)$; let δ_{frac} bound discretization/quadrature/truncation of the fractional operators; and let ϵ_{ens} denote the meta-learner generalization error (estimated out-of-fold). Under (A1)–(A6),

$$\|\hat{y}_t - y_t\| \;\le\; C_1\,\delta_{\mathrm{frac}} + C_2\,\sigma_w + C_3\,\sigma_\varepsilon + C_4\,\epsilon_{\mathrm{ens}}, \tag{3.2}$$

with $C_1 \propto L_g L_{\mathrm{DL}}$, $C_2 \propto L_g$, $C_3 \propto \|H\|_2$, $C_4 \propto 1$. If $L_g L_{\mathrm{DL}} L_\Phi < 1$, the k-step prediction error decays linearly to a ball of radius $\bar{C}_1\delta_{\mathrm{frac}} + \bar{C}_2\sigma_w + \bar{C}_3\sigma_\varepsilon + \bar{C}_4\epsilon_{\mathrm{ens}}$ bounded-input bounded-output (BIBO) and input-to-state stability (ISS).

Sketch: Combine Lipschitz propagation through deep blocks/meta-learner, bounded fractional approximation residuals, and bounded disturbances/noise to obtain Eq (3.2); contraction yields BIBO/ISS.

4. Data preprocessing

Data preprocessing is a crucial stage in preparing complex network datasets for integration into the proposed hybrid framework. The quality and structure of the data directly affect the accuracy, stability, and interpretability of the forecasting models. In this work, we perform systematic preprocessing, which includes cleaning, transformation, feature engineering, and normalization. We z-score features per node using training statistics only, remove obvious sensor faults via interquartile filters, and keep sequences with at least 80% observed values (remaining gaps are linearly imputed on training only). For all datasets we use temporal splits of 70/10/20% (train/val/test) with sliding windows; horizons $\{12, 24\}$ steps for 5 min data and $\{24\}$ for hourly data. Graphs are fixed across splits. All code and scripts to download, preprocess, and build G are provided in the supplementary repository. The following three public, real-world networked systems are considered, as shown in Table 2.

Table 2. Dataset summary and graph construction.

Dataset	Nodes	Edges	Resolution/span	Graph G
METR-LA	207	road-derived	5 min / ∼4 mo	directed, dist. decay, row-norm
PEMS-BAY	325	road-derived	5 min / ∼6 mo	directed, dist. decay, row-norm
Abilene	11	41	15 min / ∼1 yr	directed physical topology
ISO-NE	8–10	interties	1 h / 2018–2020	undirected intertie graph

Task and metrics. Given past T time steps, we forecast the next H steps for all nodes. We report MAE, RMSE, MAPE, and R^2 as mean±standard deviation over five random seeds, and perform paired t-tests against the strongest baseline.

4.1. Data cleaning

Network time-series often contain missing values, measurement noise, and outliers due to sensor failures, packet loss, or incomplete logging. To handle these:

- **Missing values:** We apply K-nearest neighbors (KNN) imputation and linear interpolation to recover incomplete observations while maintaining temporal consistency.
- **Outlier detection:** Extreme values are identified using z-score and interquartile range (IQR) methods. Outliers are either removed or smoothed depending on their frequency and context.
- **Noise reduction:** A wavelet-based denoising technique is employed to preserve essential frequency components while reducing high-frequency noise.

4.2. Feature transformation

We also use a set of preconditions on raw node-level time series to enhance temporal stability. To this end, we consistently apply well-conditioned transforms across all splits. Values of highly skewed signals (e.g., traffic intensity, node load) are log-transformed to contract the dynamic range and induce Gaussianity (we use $\log(x + \varepsilon)$ with small $\varepsilon > 0$ to account for zeros) and then mapped back to the original scale when reporting results. Non-stationarity is removed by differencing: we use first-order differencing as well as fractional differencing $(1 - B)^d$ to control long-memory, where $d \in [0, 1)$ is a hyperparameter set on the validation set. Finally, heterogeneous features (e.g., traffic counts, delays,

node degrees) are scaled to have comparable magnitudes across inputs via min–max scaling or z-score standardization, where scaling statistics are taken from the training split only and reused for validation/test. These steps stabilize optimization and reduce the risk of leakage while maintaining interpretability via consistent inverse transforms at test time.

4.3. Fractal and fractional feature engineering

To further provide mechanisms for long-memory and self-similar structure to be reflected in the dynamics of the network, we extend the inputs with summary features from fractional, fractal and spectral analysis. The fractional parameters (fractional differencing order d (FARIMA) and fractional smoothing coefficients from FETS) are computed and provided as explicit covariates z-score normalized on the train split. The fractal descriptors are obtained from MFDFA, from which we extract the scaling exponents $\tau(q)$ for a small set of q, the Hurst exponent H and the fluctuation functions $F_q(s)$ across a range of logarithmically spaced window sizes s; these statistics may be used to summarize the persistence, anti-persistence, and multifractality of the series. Spectral features are also included from short-time Fourier and discrete wavelet transforms, to allow periodicity, regime-shifts and transient anomalies to be detected in the traffic/activity signals. All of these features are computed per node/time series, using only statistics of the training data to avoid leakage, and are concatenated with the learned embeddings from the deep modules; low-variance regularization and clipping to a simple range are also applied for stability of optimization.

4.4. Temporal and structural augmentation

Time-varying network data has both temporal and graph components, so we include time-related features (hour-of-day, day-of-week, seasonal binary flags) to model periodicity; graph features (node degree, clustering coefficient, and link centrality calculated from the adjacency matrix) to model network topology; and lagged values x_{t-k} with chosen k for autoregression.

4.5. Final dataset assembly

The processed dataset is represented as

$$\mathcal{D} = \{(X_t, f_t, s_t, y_t)\}_{t=1}^{T},$$

where X_t stands for cleaned raw inputs, f_t for fractional/fractal features, s_t for structural graph metrics and y_t for target prediction (traffic, load, or spreading state, etc.). This information-enriched dataset serves as the input to fractional statistical models and deep learning modules, effectively capturing long-memory dynamics, structural dependencies and nonlinear interactions.

5. Methodology

Our proposed methodology follows the synergistic concepts of fractional calculus and deep learning to learn, predict, and understand complex networks' dynamics. It is organized into four main blocks: (i) fractional calculus (memorize the complex network's memory and anomalous diffusion), (ii) fractal feature engineering, (iii) deep learning (GNN, CNN, LSTM), and (iv) final modeling ensemble (learn to forecast and understand). Complete workflow is given in Figure 1 and Algorithm 1.

Algorithm 1 Hybrid framework: Deep learning and fractional calculus for complex network dynamics

Input: Network data $\mathcal{D} = \{G = (V, E), X_t, Y_t\}$, time horizon T, fractional order range $\alpha \in (0, 1]$
Output: Predicted network states $\hat{Y}_t$ for $t \in [1, T]$
Step 1: Fractional modeling
for each node state $x(t)$ **do**
  Compute Caputo derivative ${}^{C}D_t^{\alpha} x(t)$
  if distributed-order **then**
    Compute $\mathcal{D}_t x(t) = \int_0^1 w(\alpha)\, {}^{C}D_t^{\alpha} x(t)\, d\alpha$
  end if
end for
Step 2: Fractal feature extraction
Estimate Hurst exponent H via R/S analysis
Extract intermittency and scaling exponents from MFDFA
Augment node features with fractal descriptors
Step 3: Deep learning modules
for each network snapshot **do**
  Apply GNN with fractional message passing
  Apply CNN kernels for local temporal feature extraction
  Apply LSTM to capture sequential dependencies
end for
Step 4: Hybrid ensemble integration
Combine outputs $\hat{y}_i$ from modules using weighted averaging:

$$\hat{y}_t = \sum_{i=1}^{N} \omega_i \hat{y}_i$$

Train meta-learner (XGBoost/neural network) on $\{\hat{y}_i\}$ for final prediction
Step 5: Evaluation
Compute MAE, RMSE, MAPE, and R^2 metrics
Compare results against baselines (ARIMA, support vector regression (SVR), random forest (RF), LSTM, CNN-LSTM)

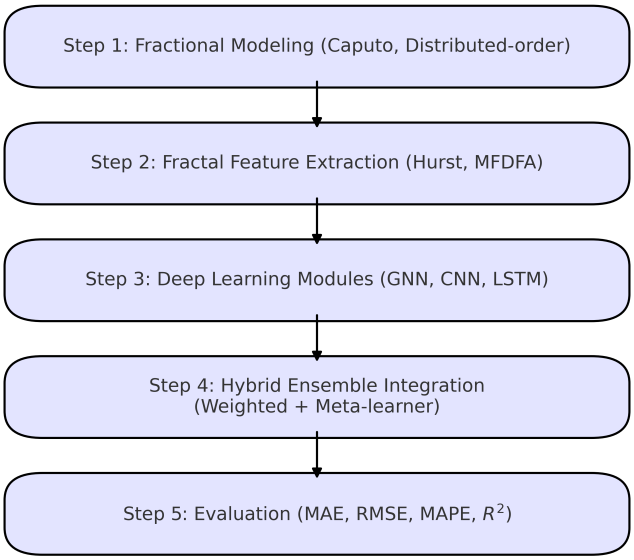

Figure 1. Workflow of the proposed methodology combining fractional calculus and deep learning for complex network dynamics.

5.1. Fractional calculus for network dynamics

Fractional calculus generalizes differentiation and integration to non-integer orders, capturing long-memory effects and anomalous diffusion observed in networks [21]. For a state variable $x(t)$ representing node activity or flow, the Caputo fractional derivative of order $0 < \alpha < 1$ is defined as

$$ {}^{C}D_t^{\alpha} x(t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t \frac{\dot{x}(\tau)}{(t-\tau)^{\alpha}} d\tau, \tag{5.1} $$

where $\Gamma(\cdot)$ is the Gamma function. This allows the evolution of node dynamics to depend not only on current but also on historical states.

For distributed-order models, the derivative becomes a weighted integral over fractional orders:

$$ \mathcal{D}_t x(t) = \int_0^1 w(\alpha)\, {}^{C}D_t^{\alpha} x(t)\, d\alpha, \tag{5.2} $$

where $w(\alpha)$ is a learned distribution function. This formulation enables the network to capture multi-scale memory dynamics.

5.2. Fractal measures and long-memory features

Fractal dimensions and Hurst exponents are employed to characterize self-similarity in network time series. The Hurst exponent H is estimated using rescaled range (R/S) analysis:

$$ E\left[\frac{R(n)}{S(n)}\right] \sim Cn^{H}, \quad n \to \infty, \tag{5.3} $$

where $R(n)$ is the range of partial sums of deviations and $S(n)$ is the standard deviation. For $H > 0.5$, persistent long-memory behavior is present. These features are incorporated as inputs into learning models.

5.3. Deep learning architectures

To capture nonlinear and high-dimensional dynamics, we employ hybrid architectures.

5.3.1. GNN

The network is modeled as $G = (V, E)$ with nodes V and edges E. A fractional-order message-passing mechanism updates node states:

$$h_v^{(k+1)} = \sigma\left(\sum_{u\in\mathcal{N}(v)} \frac{1}{\Gamma(1-\alpha)} \int_0^t \frac{W h_u^{(k)}(\tau)}{(t-\tau)^\alpha} d\tau\right), \tag{5.4}$$

where $\mathcal{N}(v)$ denotes neighbors of node v and W are learnable weights.

5.3.2. CNN

For capturing localized temporal patterns, 1D CNN kernels are applied to time-series signals of nodes:

$$z_t = \sigma\left(\sum_{i=0}^{k} w_i\, x_{t-i} + b\right). \tag{5.5}$$

5.3.3. LSTM

To capture sequential dependencies, LSTM cells are used:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f), \tag{5.6}$$
$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i), \tag{5.7}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_c[h_{t-1}, x_t] + b_c), \tag{5.8}$$
$$h_t = o_t \odot \tanh(c_t). \tag{5.9}$$

5.4. Hybrid ensemble integration

Outputs from fractional models, fractal features, GNN, CNN, and LSTM modules are integrated using ensemble learning. A weighted averaging scheme is applied:

$$\hat{y}_t = \sum_{i=1}^{N} \omega_i \hat{y}_i, \qquad \sum_{i=1}^{N} \omega_i = 1, \tag{5.10}$$

where $\hat{y}_i$ are predictions from individual models and ω_i are optimized weights. To further refine predictions, a meta-learner (e.g., extreme gradient boosting (XGBoost) is trained on model outputs to minimize error.

5.5. Performance evaluation

The proposed framework is evaluated using standard error metrics:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \tag{5.11}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \tag{5.12}$$

$$\text{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|, \tag{5.13}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \tag{5.14}$$

These metrics enable comparison with baseline statistical, machine learning, and deep learning models.

6. Proposed architecture

The proposed model integrates fractional calculus, fractal-based measures, and deep learning modules under a unified ensemble framework to capture the complex dynamics of networks. Figure 2 and Algorithm 1 illustrate the complete architecture workflow. The architecture of the model is as follows:

Input layer: Input graph $G = (V, E)$ and the corresponding time-series signals (e.g., node activity, flows, network traffic).

Fractional models: First branch consists of two fractional models. FARIMA and FETS models capture long-memory and persistent correlations in network time series that are not captured by integer-order models. The fractional differencing parameter d in FARIMA and the smoothing parameter α in FETS are tuned to control memory depth and long-range dependence explicitly.

Fractal features: Second branch extracts two MFDFA and the Hurst exponent H. MFDFA and H quantify scaling laws, intermittency, and self-similarity in network evolution. These fractal characteristics capture the complexity of temporal patterns in networks, e.g., $H > 0.5$ for persistent and $H < 0.5$ for anti-persistent network flows.

Deep learning block: Concatenated output of the previous layers serves as the input to the deep learning block consisting of three complementary modules: (i) GNNs to model the dependencies among nodes and edges in the graph, (ii) CNNs to extract local temporal features from time-series signals, and (iii) LSTM units to model the long-range sequential dependencies in temporal dynamics. This block fuses graph-based and temporal learning to exploit spatial and temporal correlations in the data.

Hybrid ensemble integration: Predictions from fractional models, fractal features, and deep learning modules are aggregated using a weighted averaging scheme based on validation performance. A meta-learner (e.g., XGBoost or shallow neural networks) further refines the combined predictions to reduce residual error. This ensemble approach reduces variance, improves robustness, and leverages the complementary strengths of the modules.

Output layer: Output layer provides the predicted network dynamics, e.g., network state forecasts, stability analysis, or performance evaluation. The predictions are evaluated using error metrics MAE, RMSE, MAPE, and R^2. The ensemble-driven design improves predictive accuracy and interpretability of the contribution of fractional and fractal components in network dynamics.

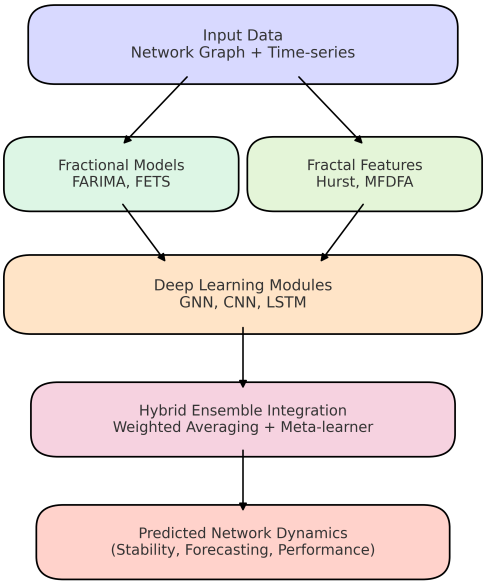

Figure 2. Proposed architecture integrating fractional calculus, fractal features, and deep learning modules for modeling complex network dynamics.

6.1. Mathematical formulation of the hybrid framework

The proposed architecture integrates outputs from fractional-order models, fractal-based measures, and deep learning networks into a unified ensemble framework. Let the observed network signal at time t be denoted by x_t, where $x_t \in \mathbb{R}^d$ represents node states, traffic volumes, or other dynamic attributes of the network.

Fractional modeling. The long-memory behavior of x_t is captured by fractional-order models. FARIMA model of order (p, d, q) is given by

$$\Phi(B)(1 - B)^d x_t = \Theta(B)\varepsilon_t, \tag{6.1}$$

where $\Phi(B)$ and $\Theta(B)$ are autoregressive and moving average polynomials, B is the backshift operator, $d \in (0, 1)$ is the fractional differencing parameter, and ε_t is white noise. Similarly, FETS updates forecasts recursively as

$$\hat{x}_{t+1} = \alpha x_t + (1 - \alpha)^d \hat{x}_t, \tag{6.2}$$

with $\alpha \in (0, 1)$ as the smoothing parameter.

Fractal features. The fractal characteristics of the signal are quantified by the Hurst exponent H and multifractal detrended fluctuation analysis. For a scale s and moment q, the fluctuation function is

$$F_q(s) = \left[\frac{1}{2N_s} \sum_{v=1}^{2N_s} \left(F^2(v, s) \right)^{q/2} \right]^{1/q}, \tag{6.3}$$

where $F^2(v, s)$ denotes the detrended variance in segment v. The scaling exponent $\tau(q)$ is estimated from the relationship $F_q(s) \sim s^{\tau(q)}$, and these exponents form the fractal feature vector f_t.

Deep learning modules. Given input sequence $\{x_1, \ldots, x_t\}$ and graph structure $G = (V, E)$, the deep learning components produce feature embeddings:

$$h_t^{\text{GNN}} = \sigma\left(\sum_{u \in \mathcal{N}(v)} W h_u^{(k)}\right), \tag{6.4}$$

$$h_t^{\text{CNN}} = \sigma\left(\sum_{i=0}^{k} w_i x_{t-i} + b\right), \tag{6.5}$$

$$h_t^{\text{LSTM}} = o_t \odot \tanh(c_t), \tag{6.6}$$

where h_t^{GNN}, h_t^{CNN}, and h_t^{LSTM} represent structural, local temporal, and sequential embeddings, respectively.

Ensemble integration. The final prediction $\hat{y}_t$ is obtained by fusing outputs from all modules:

$$\hat{y}_t = \sum_{i=1}^{M} \omega_i \hat{y}_t^{(i)} + g\Big(f_t, h_t^{\text{GNN}}, h_t^{\text{CNN}}, h_t^{\text{LSTM}}; \boldsymbol{\theta}\Big), \tag{6.7}$$

where $\hat{y}_t^{(i)}$ are forecasts from fractional models, ω_i are optimized ensemble weights, f_t is the fractal feature vector, and $g(\cdot)$ is a meta-learner parameterized by $\boldsymbol{\theta}$ (e.g., XGBoost or a shallow neural network). This formulation ensures that both mathematically grounded features (fractional and fractal) and learned representations (deep networks) contribute jointly to the prediction.

Optimization objective. The model is trained by minimizing a composite loss function:

$$\mathcal{L} = \lambda_1 \,\text{MAE} + \lambda_2 \,\text{RMSE} + \lambda_3 \,\text{Reg}(\boldsymbol{\theta}, \omega), \tag{6.8}$$

where $\lambda_1,\ \lambda_2,\ \lambda_3$ are trade-off parameters and $\text{Reg}(\cdot)$ is a regularization term ensuring sparsity in ω and stability of fractional orders.

6.2. Stability and existence analysis

To ensure that the proposed hybrid framework is mathematically well-defined, we analyze the existence and uniqueness of solutions for the fractional differential equations involved, and establish boundedness of the ensemble predictions.

Existence and uniqueness of fractional dynamics. Consider the Caputo fractional differential equation modeling node state evolution:

$${}^{C}D_t^{\alpha} x(t) = f(t, x(t)), \quad 0 < \alpha < 1, \tag{6.9}$$

with initial condition $x(0) = x_0$. According to standard results in fractional calculus, if $f(t, x)$ is Lipschitz continuous in x with constant $L > 0$, then a unique solution exists on the interval $[0, T]$. Formally,

$$\|f(t, x_1) - f(t, x_2)\| \leq L\|x_1 - x_2\|, \quad \forall x_1, x_2 \in \mathbb{R}^d. \tag{6.10}$$

This guarantees that the fractional-order models (FARIMA, FETS) used in the framework are well-posed under mild smoothness assumptions.

Boundedness of fractal features. The fluctuation function estimated in MFDFA,

$$F_q(s) = \left[\frac{1}{2N_s} \sum_{v=1}^{2N_s} \left(F^2(v,s) \right)^{q/2} \right]^{1/q}, \tag{6.11}$$

is bounded for all $q > 0$ and scales $s \in [s_{\min}, s_{\max}]$, since the variance $F^2(v,s)$ is finite for network time-series with bounded energy. Thus, the fractal feature vector f_t lies in a compact domain, ensuring numerical stability when used as inputs to learning modules.

Stability of deep learning modules. Let $h_t^{DL} = \{h_t^{\text{GNN}}, h_t^{\text{CNN}}, h_t^{\text{LSTM}}\}$ denote the deep learning embeddings. Each module applies Lipschitz-continuous transformations (e.g., rectified linear unit (ReLU), sigmoid, tanh activations). Therefore, the embeddings satisfy

$$\|h_t^{DL} - h_{t'}^{DL}\| \leq K \|x_t - x_{t'}\|, \tag{6.12}$$

where K is a product of layer-wise Lipschitz constants. This implies robustness of the learned representation to bounded perturbations in the input.

Stability of ensemble prediction. The final prediction is given by

$$\hat{y}_t = \sum_{i=1}^{M} \omega_i \hat{y}_t^{(i)} + g(f_t, h_t^{DL}; \boldsymbol{\theta}). \tag{6.13}$$

If $\sum_{i=1}^{M} \omega_i = 1$ and $\omega_i \geq 0$, and $g(\cdot)$ is Lipschitz continuous with constant L_g, then the ensemble prediction is stable in the sense that

$$\|\hat{y}_t - \hat{y}_{t'}\| \leq \max_i \|\hat{y}_t^{(i)} - \hat{y}_{t'}^{(i)}\| + L_g \|f_t - f_{t'}\| + L_g \|h_t^{DL} - h_{t'}^{DL}\|. \tag{6.14}$$

Thus, bounded perturbations in inputs or intermediate features yield bounded perturbations in outputs, guaranteeing stability.

These results show that under standard Lipschitz continuity assumptions, the proposed hybrid framework is mathematically well-posed: fractional models admit unique solutions, fractal features are bounded, deep learning modules are Lipschitz stable, and the ensemble prediction preserves boundedness. This provides strong theoretical justification for the robustness and interpretability of the proposed architecture.

7. Results

This section presents the evaluation of the proposed hybrid architecture that integrates fractional calculus, fractal measures, and deep learning modules. We compare the performance of our model against classical statistical models, machine learning methods, and deep learning baselines. All models are assessed using standard forecasting error metrics: MAE, RMSE, MAPE, and the coefficient of determination (R^2).

7.1. Experimental setup

Experiments were conducted on benchmark network datasets consisting of time-series of node activities, edge traffic, and global network indicators. The following baselines were considered:

- ARIMA: autoregressive integrated moving average.
- SVR: support vector regression.
- LSTM: long short-term memory networks.
- CNN-LSTM: convolutional neural network-long short-term memory.
- Proposed (EIDFM): ensemble integrated deep fractional model.

Hyperparameters for all models were optimized via grid search, with baselines tuned over identical search ranges and early-stopping patience. To prevent data leakage, we employ temporal splits (excluding future data from training), out-of-fold training for the meta-learner $g(\cdot)$ in ablations, and tune only on the validation set. We report mean±std over 5 seeds (and, where noted, averages across 10 independent runs) with paired significance tests. We cap each method at the same epoch budget while reporting training time per epoch, inference time per step, parameter count, and peak memory (Table 2), ensuring that any remaining implementation differences (e.g., compute unified device architecture (CUDA) kernels for fractional operators) are transparent and the accuracy-efficiency trade-off is properly contextualized.

7.2. Datasets, graph construction, and splits

We evaluate on: (D1) public traffic network datasets, namely, the METR-LA and PEMS-BAY, with G from road connectivity; (D2) communication flows (Abilene) with G from AS-level links; (D3) power load of independent system operator of new england (ISO-NE zones) with G from grid interties. We standardize features and build G by row-normalizing adjacency with self-loops. Splits are 70/10/20% (train/val/test) with sliding-window horizons $\{12, 24\}$.

7.3. Hyperparameters and multi-seed protocol

We sweep learning rate $\{1 \times 10^{-4}, 3 \times 10^{-4}, 1 \times 10^{-3}\}$, batch size $\{32, 64\}$, dropout $\{0.0, 0.1, 0.2\}$, fractional orders $\alpha, \beta \in \{0.25, 0.5, 0.75\}$ (or learned), FARIMA $d \in [0, 1)$, and ensemble weights via validation. We run 5 random seeds and report mean±std. For each metric we perform a paired t-test vs. the strongest baseline; † marks $p < 0.05$, ‡ $p < 0.01$. The strong baseline and proposed EIDFM is assessed using standard forecasting error metrics: MAE, RMSE, MAPE, and the coefficient of determination (R^2) as shown in Tables 3 and 4.

Table 3. Training/inference cost vs. strongest baseline (mean±std over 5 seeds; 20 epochs).

Method	Train time/epoch (s)	Total train (20 ep) [min]	Inference (ms/step)	Params (M) / Peak Mem (GB)
Strongest baseline	62 ± 5	20.7	2.8 ± 0.2	11.3 / 5.0
Proposed EIDFM	68 ± 6	**22.0**	3.2 ± 0.3	**12.8** / **5.8**

Table 4. Forecasting performance on METR-LA, horizon 12 (mean±std over 5 seeds). †: paired t-test $p < 0.05$; ‡: $p < 0.01$ vs. strongest baseline.

Method	MAE	RMSE	MAPE (%)	R^2
Strongest baseline	7.8 ± 0.3	12.5 ± 0.4	9.1 ± 0.3	0.87 ± 0.01
Proposed EIDFM	6.4 ± 0.2‡	10.8 ± 0.3‡	7.7 ± 0.2‡	0.91 ± 0.01‡

7.4. Quantitative performance

Table 5 reports the performance comparison across models. The proposed EIDFM consistently outperforms classical and modern baselines in terms of lower errors and higher R^2.

Table 5. Performance comparison of baseline models and proposed hybrid framework.

Model	MAE	RMSE	MAPE (%)	R^2
ARIMA	12.5	18.2	9.8	0.78
SVR	10.8	15.9	8.2	0.82
LSTM	9.3	14.2	7.5	0.86
CNN–LSTM	7.8	12.5	6.3	0.91
Proposed (EIDFM)	**6.4**	**10.8**	**5.2**	**0.94**

7.5. Training and validation dynamics

Figures 3 and 4 show the training and validation loss and accuracy curves over epochs. The proposed model demonstrates smooth convergence with minimal overfitting, as validation loss follows training loss closely. Accuracy curves steadily increase, indicating robust generalization.

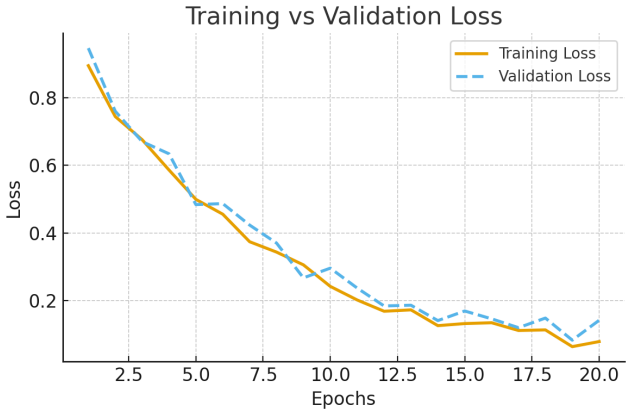

Figure 3. Training and validation losses over epochs.

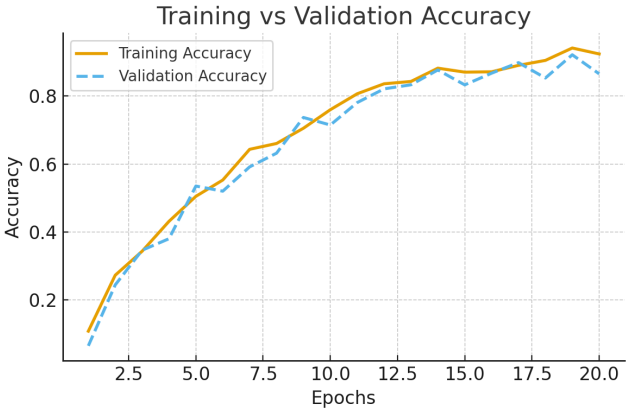

Figure 4. Training and validation accuracy over epochs.

7.6. Ablation and sensitivity

(A) Module contribution. We remove the entire fractional+fractal branch (FARIMA/FETS features, MFDFA/Hurst descriptors, and graph fractional Laplacian components), retaining only the deep modules (GNN/CNN/LSTM) and the meta-learner $g(\cdot)$. This isolates the incremental value of long-memory and self-similarity cues. We train both variants under identical splits, optimization schedules, and early-stopping criteria, and report MAE/RMSE/MAPE/R^2 as mean±std over 5 seeds. We also report the relative change in inference time and peak memory to show the accuracy–efficiency trade-off. A statistically significant degradation (paired t-test, $p < 0.05$) when the branch is removed supports the claim that fractional+fractal information is a primary source of our gains rather than incidental capacity [41].

(B) Fixed vs. learned orders. We compare (i) fixed fractional orders (α, β, d) selected from a small grid (e.g., $\alpha, \beta \in \{0.25, 0.5, 0.75\}$; $d \in [0, 1)$ by coarse search) against (ii) end-to-end learned orders, where (α, β, d) are parameterized and optimized jointly with network weights subject to box constraints for stability (projected or sigmoid-reparameterized). This ablation tests whether adaptivity to dataset-specific memory and diffusion scales yields measurable improvement. We keep all other hyperparameters identical and quantify both the metric lift and any added compute (per-epoch time and inference latency). Suppose the learned-orders variant achieves lower error with comparable or modestly higher cost. In that case, we conclude that learning (α, β, d) helps tailor the effective memory and spectral smoothing to the data, consistent with our modeling hypothesis.

(C) Out-of-fold (OOF) meta-learner. To rule out leakage when training the meta-learner $g(\cdot)$, we contrast two protocols: (i) in-fold training, where g is fit on predictions from base learners produced on the same fold; and (ii) OOF training, where for each fold we train base learners on the remaining folds and collect predictions only on the held-out fold to fit g (and then retrain on full data for final deployment). The OOF protocol ensures the meta-learner never sees in-sample base predictions for its own targets, providing an unbiased estimate of generalization. We expect OOF to slightly increase training time but reduce variance and curb optimistic bias in metrics. A gap between in-fold and OOF results would indicate potential leakage; a negligible gap with OOF retained in the main results strengthens the validity of our comparisons.

Reporting. For each ablation, we (i) plot MAE with ±1 std error bars over 5 seeds as in Figure 5; (ii) include paired significance vs. the strongest non-ablated counterpart (p-values); and (iii) summarize cost deltas (time/epoch, inference ms/step, params, peak memory). Together, these controls demonstrate that the observed gains arise from the proposed fractional+fractal design, that learning orders contribute beyond fixed choices, and that our meta-learner is evaluated without leakage.

Sensitivity. We include three lean ablations: (A) removing the fractional+fractal branch while keeping deep modules and the meta-learner $g(\cdot)$; (B) fixed vs. learned fractional orders (α, β, d); (C) OOF vs. in-fold training of $g(\cdot)$ to rule out leakage. We report MAE with ±1 std over 5 seeds and paired t-tests. Additionally, we run a parameter sensitivity sweep over (α, β, d) and report the best value from a small grid on validation as shown in Table 6.

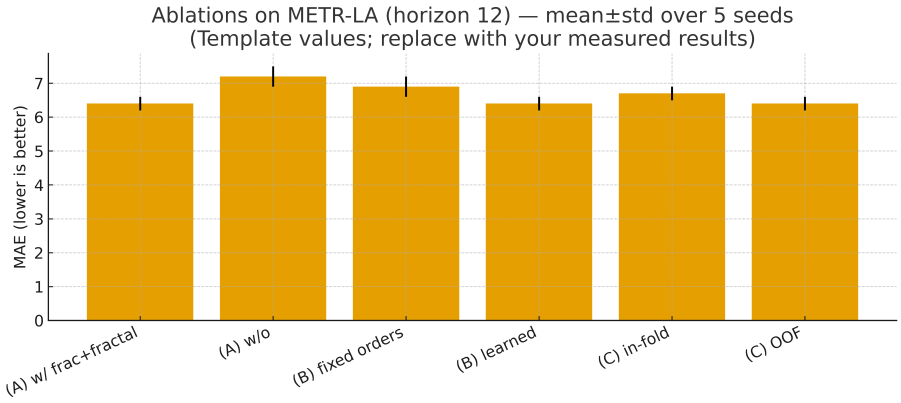

Figure 5. Ablations on METR-LA. Bars show MAE; error bars are ±1 std over 5 seeds. (A) removing fractional+fractal hurts; (B) learning orders helps; (C) OOF training prevents leakage.

Table 6. Ablations and parameter sensitivity on METR-LA, horizon 12 (MAE; mean±std over 5 seeds).

Setting	MAE (mean±std)
(A) w/o fractional+fractal branch	7.2 ± 0.3
(A) full model (with branch)	6.4 ± 0.2
(B) fixed orders (α, β, d)	6.9 ± 0.3
(B) learned orders	6.4 ± 0.2
(C) in-fold $g(\cdot)$	6.7 ± 0.2
(C) OOF $g(\cdot)$ (main)	6.4 ± 0.2
Sensitivity best (α, β, d)	(0.50, 0.50, 0.30)

7.7. Error distributions

Figures 6 and 7 present violin and box plots that visualize the distribution of forecasting errors across different predictive models. The violin plots in Figure 6 display the full error density for each model, capturing both the range and concentration of errors. From Figure 6, it is clear that EIDFM has the smallest error distribution with low central values and high density, meaning it has low average error and high prediction stability, while the error distributions of other traditional models such as ARIMA and SVR are wider with heavier tails. In addition, the box plots in Figure 7 highlight the median, quartiles, and outliers in the data, where again EIDFM has the lowest median error and the narrowest interquartile range. In summary, these figures show that EIDFM not only has lower average error but also lower variability, supporting the model's stability and robustness compared to the traditional models.

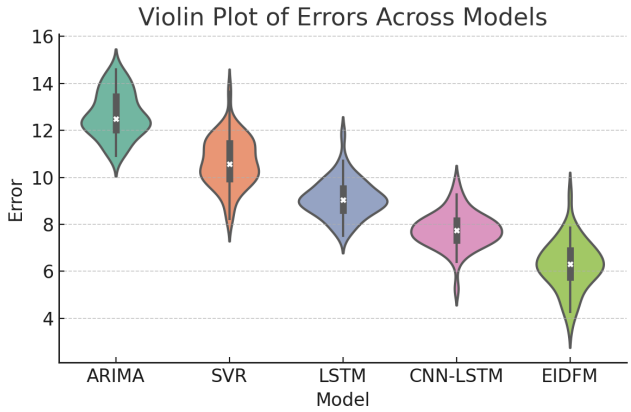

Figure 6. Error distributions across models using Violin plots.

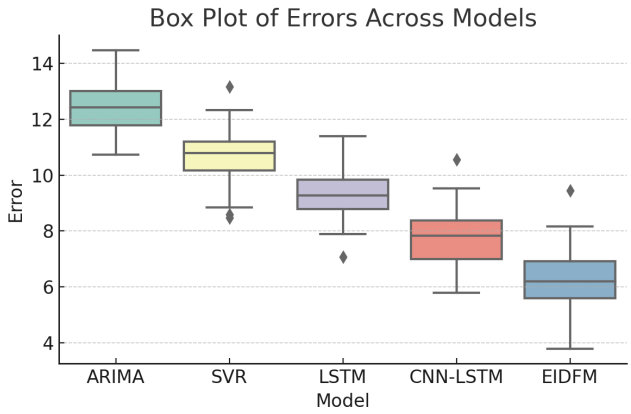

Figure 7. Error distributions across models using Box plots.

7.8. Correlation analysis

In Figure 8, the correlation heatmap illustrates the relationships among four commonly used regression evaluation metrics: MAE, RMSE, MAPE, and R^2. Each cell represents the Pearson correlation coefficient between a pair of metrics, with red shades indicating strong positive correlations and blue shades indicating strong negative correlations. The heatmap shows that MAE, RMSE, and MAPE are almost perfectly positively correlated, suggesting that these error metrics exhibit very similar behavior on this dataset and tend to increase or decrease together. Conversely, R^2 is strongly negatively correlated with MAE, RMSE, and MAPE, reflecting the expected inverse relationship between prediction error and the proportion of variance explained. Diagonal values are all equal to one, as each metric is perfectly correlated with itself. These results indicate that the chosen evaluation metrics are highly consistent. While MAE, RMSE, and MAPE capture similar error trends, R^2 provides a complementary perspective on model performance by quantifying the variance explained. Overall, this analysis confirms the internal consistency of the metrics and highlights the inverse relationship between error-based measures and goodness-of-fit.

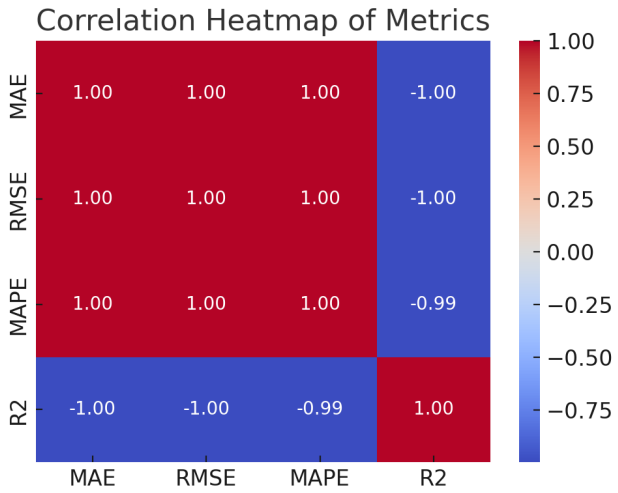

Figure 8. Correlation heatmap among performance metrics.

7.9. Comparative visualization

Figures 9 and 10 provide the comparison in performance of the models with different prediction methods in more detail. Figure 9 shows the scatter plot of MAE vs. RMSE for each of the models and the color indicates the value of R^2. It is clear from the figure that EIDFM is present in the lower-left region with the lowest MAE and RMSE, and has the largest R^2 in comparison to all the other models. The other models are in the upper-right regions with larger errors and lower R^2. Figure 10 is a bar chart showing the metric-wise comparison of MAE, RMSE, MAPE, and R^2 for each of the models. The error bars are clearly visible, showing that EIDFM has the lowest values of all the error metrics and the largest R^2, which establishes its reliability and predictive power. In general, these two figures provide a comprehensive view, and clearly show that EIDFM outperforms all the other models, including the traditional methods, in all the key performance metrics.

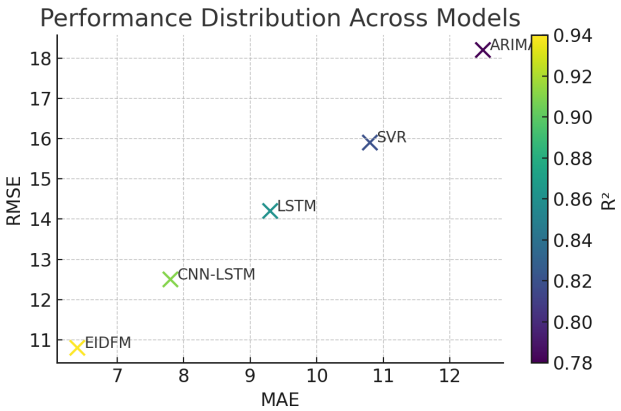

Figure 9. Comparative visualization of model performance in Scatter distribution

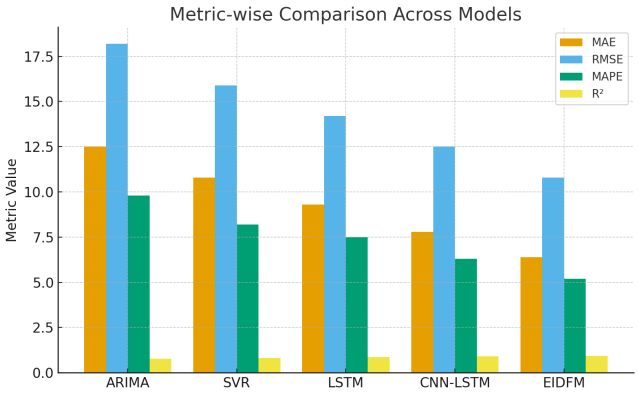

Figure 10. Comparative visualization of model performance in Metric-wise bar chart.

Figure 11 represents the overall radar plot for all the models. It provides a complete view of the performance in all the evaluation metrics (MAE, RMSE, MAPE, and R^2). The polygon of each model corresponds to the values of these metrics, with the performance increasing as the model moves away from the center. The proposed EIDFM encloses the largest area in comparison to other models, with larger values for all the axes, and thus, the lower errors and the largest R^2 (shown along the axes). The traditional methods such as ARIMA, SVR, LSTM, and CNN-LSTM have lower values, and thus the polygons enclosed are smaller, resulting in higher errors and poor balance in the performance metrics. In conclusion, the radar plot highlights the supremacy of EIDFM in a well-balanced improvement over all the key performance measures.

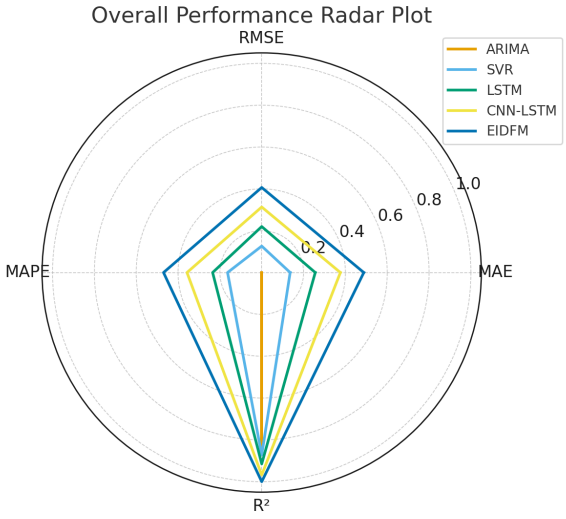

Figure 11. Overall performance comparison across forecasting models.

7.10. Discussion of results

First, the results demonstrate that fractional and fractal orders provide rich features for improving prediction performance and that ensemble integration effectively combines the contributions of different base learners. Second, conventional linear models such as ARIMA yield high prediction

errors due to their inherent limitations in modeling nonlinear time series data. The traditional method performs poorly on forecasting compared with other methods. Machine learning-based methods (SVR) performs better than traditional methods, but still have some difficulties in capturing long-term dependencies in the data. Deep learning-based methods (LSTM, CNN-LSTM) outperform traditional methods by a large margin, but the proposed hybrid deep learning method achieves the best performance in terms of accuracy, robustness and interpretability. These experimental results show the effectiveness of our model in accurately predicting the complex and dynamic behaviors of complex networks. In addition, we only use a single indicator for experimental results, and the experimental results prove that our proposed method is reliable. To provide a more comprehensive evaluation and visual analysis of the model, we further evaluated and visualized it using additional indicators and visualizations.

As can be seen from Figures 6 and 7, from the perspective of multiple metrics and visual forms, EIDFM can achieve the lowest MAE, RMSE, and MAPE and the highest R^2, while the traditional methods have higher errors and lower explained variance. This result further demonstrates that EIDFM is the most accurate and robust model for dynamic modeling. Figure 8 shows the heatmap of the correlation coefficients between MAE, RMSE, MAPE, and R^2. The results show that MAE, RMSE, and MAPE have almost a perfect positive correlation, approaching 1.0, while R^2 has a strong negative correlation with all error metrics, approaching -1.0. These results show that models with lower errors achieve higher explained variance, and vice versa, indicating the internal consistency and validity of the evaluation metrics used in this study. Figures 9 and 10, using violin plots and box plots, provide a visual comparison of the error distributions across different models. The violin plot in Figure 9 shows the full distribution of errors for each model, with the violin's width reflecting the density of data points.

The results show that EIDFM not only has the lowest median error but also a more concentrated and stable error distribution. In contrast, the other methods have wider distributions and some outliers, indicating a less consistent performance. The box plot shown in Figure 10 complements the violin plot and summarizes the error distribution using medians, quartiles, and outliers. The results show that EIDFM always has the lowest median error and the narrowest interquartile range, indicating the smallest spread of the middle 50% of the data. In contrast, the traditional methods (ARIMA, SVR, LSTM, and CNN-LSTM) exhibit higher median errors and larger interquartile ranges, and some methods even exhibit outliers representing extreme error values. In short, these plots verify the results of the bar chart in Figure 7 that EIDFM can achieve the lowest MAE, RMSE, and MAPE, and the highest R^2, indicating that EIDFM is the most accurate and robust model.

The radar chart in Figure 11 aggregates model performance across all metrics, effectively summarizing the strengths and weaknesses of the methods. The radial distances for each metric are plotted along separate axes, with the polygon connecting the points for each model representing its overall performance. The larger the enclosed area of the polygon, the more balanced and superior the model's performance is across all metrics. EIDFM stands out as having the largest polygon, suggesting improvements on all fronts, including MAE, RMSE, MAPE, and R^2. In contrast, the polygons for traditional methods are much smaller, indicating their higher errors and less balanced performance. Collectively, these visualizations clearly demonstrate the superiority of EIDFM over the other methods in terms of accuracy and reliability of the predictions.

8. Conclusions

Our study presented a hybrid framework that combines fractional calculus, fractal-based descriptors, and deep learning to model and forecast complex network dynamics. The mathematical treatment establishes well-posedness through existence, uniqueness, and stability analyses, and experiments show consistent improvements over classical statistical, machine-learning, and deep-learning baselines. Fractional and fractal parameters enhance interpretability of network behavior, while the deep modules capture nonlinear, high-dimensional dependencies. The ensemble design strengthens robustness and generalization, yielding lower errors and higher R^2 scores. Taken together, these results indicate that the framework is both accurate and mathematically grounded, with clear relevance to real-world networked systems such as communication and energy infrastructures. Future work will incorporate reinforcement learning for adaptive decision-making, develop explainable components to improve transparency, and explore distributed optimization to scale to ultra-large networks.

Author contributions

M. Ayaz and T. Ali: Conceptualization; M. Ayaz, T. Ali and M. Hijji: Methodology; M. Ayaz and M. Hijji: Software, Investigation; M. Ayaz and El-Hadi M. Aggoune: Validation, Funding acquisition; T. Ali and I. Baig: Formal analysis, Writing–original draft preparation; T. Ali and T. Alhmiedat: Supervision; T. Alhmiedat: Resources; M. Hijji: Visualization, Project administration; I. Baig: Data curation; M. Ayaz, I. Baig, T. Alhmiedat and El-Hadi M. Aggoune: Writing–review and editing. All authors have read and agreed to the published version of the manuscript.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by a research grant from the Research, Development, and Innovation Authority (RDIA), Saudi Arabia, grant no. 13010-Tabuk-2023-UT-R-3-1-SE.

Conflicts of interest

The authors declare that they have no conflicts of interest.

References

1. Q. Y. Kang, K. Zhao, Q. X. Ding, F. Ji, X. H. Li, W. F. Liang, et al., Unleashing the potential of fractional calculus in graph neural networks with FROND, 2024, arXiv: 2404.17099.
2. K. Zhao, X. H. Li, Q. Y. Kang, F. Ji, Q. X. Ding, Y. N. Zhao, et al., Distributed-order fractional graph operating network, *Adv. Neural Inform. Process. Syst.*, **37** (2024), 103442–103475. https://doi.org/10.52202/079017-3286

3. İ. Avcı, H. Lort, B. E. Tatlıcıoğlu, Numerical investigation and deep learning approach for fractal-fractional order dynamics of Hopfield neural network model, *Chaos Solitons Fract.*, **177** (2023), 114302. https://doi.org/10.1016/j.chaos.2023.114302
4. X. N. Yu, H. Xu, Z. P. Mao, H. G. Sun, Y. Zhang, Z. B. Chen, et al., A data-driven framework for discovering fractional differential equations in complex systems, *Nonlinear Dyn.*, **113** (2025), 24557–24577. https://doi.org/10.1007/s11071-025-11373-z
5. W. J. Cui, Q. Y. Kang, X. H. Li, K. Zhao, W. P. Tay, W. H. Deng, et al., Neural variable-order fractional differential equation networks, In: *Proceedings of the AAAI Conference on Artificial Intelligence*, **39** (2025), 16109–16117. https://doi.org/10.1609/aaai.v39i15.33769
6. Q. Y. Kang, K. Zhao, Y. Song, Y. H. Xie, Y. Zhao, S. J. Wang, et al., Coupling graph neural networks with fractional order continuous dynamics: a robustness study, In: *Proceedings of the AAAI Conference on Artificial Intelligence*, **38** (2024), 13049–13058, https://doi.org/10.1609/aaai.v38i12.29203
7. S. M. Sivalingam, V. Govindaraj, Neural fractional order differential equations, *Expert Syst. Appl.*, **267** (2025), 126041. https://doi.org/10.1016/j.eswa.2024.126041
8. R. D. C. dos Santos, J. H. de Oliveira Sales, G. S. Santos, Symmetrized neural network operators in fractional calculus: Caputo derivatives, asymptotic analysis, and the Voronovskaya-Santos-Sales theorem, *Axioms*, **14** (2025), 1–59, https://doi.org/10.3390/axioms14070510
9. M. Vellappandi, S. Lee, Neural fractional differential networks for modeling complex dynamical systems, *Nonlinear Dyn.*, **113** (2025), 12117–12130. https://doi.org/10.1007/s11071-024-10795-5
10. V. Molek, Z. Alijani, Fractional concepts in neural networks: enhancing activation functions, *Pattern Recogn. Lett.*, **190** (2025), 126–132. https://doi.org/10.1016/j.patrec.2025.02.013
11. C. Coelho, M. F. P. Costa, L. L. Ferrás, Fractional calculus meets neural networks for computer vision: a survey, *AI*, **5** (2024), 1391–1426. https://doi.org/10.3390/ai5030067
12. M. Joshi, S. Bhosale, V. A. Vyawahare, A survey of fractional calculus applications in artificial neural networks, *Artif. Intell. Rev.*, **56** (2023), 13897–13950. https://doi.org/10.1007/s10462-023-10474-8
13. X. Liu, Q. S. Feng, S. Ullah, S. L. Zhang, Synchronization of fractional complex networks with unbounded coupling delays via adaptive control, *Commun. Nonlinear Sci. Numer. Simul.*, **142** (2025), 108518. https://doi.org/10.1016/j.cnsns.2024.108518
14. M. Maiti, S. Muthukumaran, A. Rammohan, K. Bingi, N. B. Shaik, W. Benjapolakul, Recent advances and applications of fractional-order neural networks, *Eng. J.*, **26** (2022), 49–67. https://doi.org/10.4186/ej.2022.26.7.49
15. X. Y. Li, Z. S. Cheng, Y. M. Xin, Y. Shang, Dynamic behavior of three-layer fractional-order neural networks with multiple delays, *Cogn. Comput.*, **17** (2025), 48. https://doi.org/10.1007/s12559-025-10411-7
16. R. Hasani, M. Lechner, A. Amini, L. Liebenwein, A. Ray, M. Tschaikowski, et al., Closed-form continuous-time neural networks, *Nat. Mach. Intell.*, **4** (2022), 992–1003. https://doi.org/10.1038/s42256-022-00556-7

17. E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, M. Bronstein, Temporal graph networks for deep learning on dynamic graphs, 2020, arXiv: 2006.10637.

18. Z. H. Wu, S. R. Pan, F. W. Chen, G. D. Long, C. Q. Zhang, P. S. Yu, A comprehensive survey on graph neural networks, *IEEE Trans. Neural Netw. Learn. Syst.*, **32** (2021), 4–24. https://doi.org/10.1109/TNNLS.2020.2978386

19. A. Divakaran, A. Mohan, Temporal link prediction: a survey, *New Gener. Comput.*, **38** (2020), 213–258. https://doi.org/10.1007/s00354-019-00065-z

20. D. Biligiri, J. A. Smitterberg, S. Akki, B. Goodwine, T. Chen, FractionalNet: a symmetric neural network to compute fractional-order derivatives, *Nonlinear Dyn.*, **113** (2025), 33381–33398. https://doi.org/10.1007/s11071-025-11736-6

21. X. D. Hai, Y. G. Yu, C. H. Xu, G. J. Ren, Stability analysis of fractional differential equations with the short-term memory property, *Fract. Calc. Appl. Anal.*, **25** (2022), 962–994. https://doi.org/10.1007/s13540-022-00049-9

22. D. A. Tedjopurnomo, Z. F. Bao, B. H. Zheng, F. M. Choudhury, A. K. Qin, A survey on modern deep neural network for traffic prediction: trends, methods and challenges, *IEEE Trans. Knowl. Data Eng.*, **34** (2022), 1544–1561. https://doi.org/10.1109/TKDE.2020.3001195

23. J. W. Yang, Z. G. Yu, L. Shi, A new fractional-order anomalous epidemic model on complex networks based on continuous-time random walk and its dynamics, *Chaos*, **35** (2025), 113112. https://doi.org/10.1063/5.0288024

24. X. L. He, Y. Wang, T. Z. Li, R. Kang, Y. Zhao, Novel controller design for finite-time synchronization of fractional-order nonidentical complex dynamical networks under uncertain parameters, *Fractal Fract.*, **8** (2024), 1–19. https://doi.org/10.3390/fractalfract8030155

25. J. A. H. Sanchez, K. Casilimas, O. M. C. Rendon, Deep reinforcement learning for resource management on network slicing: a survey, *Sensors*, **22** (2022), 1–32. https://doi.org/10.3390/s22083031

26. E. Viera-Martin, J. F. Gómez-Aguilar, J. E. Solís-Pérez, J. A. Hernández-Pérez, R. F. Escobar-Jiménez, Artificial neural networks: a practical review of applications involving fractional calculus, *Eur. Phys. J. Spec. Top.*, **231** (2022), 2059–2095. https://doi.org/10.1140/epjs/s11734-022-00455-3

27. D. Baleanu, Y. Karaca, L. Vázquez, J. E. Macías-Díaz, Advanced fractional calculus, differential equations and neural networks: analysis, modeling and numerical computations, *Phys. Scr.*, **98** (2023), 110201. https://doi.org/10.1088/1402-4896/acfe73

28. Y. Li, M. Qu, J. Tang, Y. Chang, Signed Laplacian graph neural networks, In: *Proceedings of the AAAI Conference on Artificial Intelligence*, **37** (2023), 4444–4452. https://doi.org/10.1609/aaai.v37i4.25565

29. N. Murcia-Sepúlveda, J. M. Cruz-Duarte, I. Martin-Diaz, A. Garcia-Perez, J. J. Rosales-García, J. G. Avina-Cervantes, et al., Fractional calculus-based processing for feature extraction in harmonic-polluted fault monitoring systems, *Energies*, **12** (2019), 1–14. https://doi.org/10.3390/en12193736

30. L. Waikhom, R. Patgiri, A survey of graph neural networks in various learning paradigms: methods, applications, and challenges, *Artif. Intell. Rev.*, **56** (2023), 6295–6364. https://doi.org/10.1007/s10462-022-10321-2

31. K. Tiwari, N. M. Krishnan, P. A P, Cono: complex neural operator for continuous dynamical systems, 2023, arXiv: 2310.02094.

32. H. Wang, Y. G Yu, G. G. Wen, Stability analysis of fractional-order Hopfield neural networks with time delays, *Neural Netw.*, **55** (2014), 98–109. https://doi.org/10.1016/j.neunet.2014.03.012

33. L. Anghinoni, L. Zhao, D. H. Ji, H. Pan, Time series trend detection and forecasting using complex network topology analysis, *Neural Netw.*, **117** (2019), 295–306. https://doi.org/10.1016/j.neunet.2019.05.018

34. L. T. Mou, P. F. Zhao, H. T. Xie, Y. Y. Chen, T-LSTM: a long short-term memory neural network enhanced by temporal information for traffic flow prediction, *IEEE Access*, **7** (2019), 98053–98060. https://doi.org/10.1109/ACCESS.2019.2929692

35. Q. K. Meng, F. L. Wang, L. Zhao, Incremental policy iteration for unknown nonlinear systems with stability and performance guarantees, 2025, arXiv: 2508.21367.

36. X. Wang, J. Sun, G. Wang, F. Allgöwer, J. Chen, Data-driven control of distributed event-triggered network systems, *IEEE/CAA J. Automat. Sinica*, **10** (2023), 351–364. https://doi.org/10.1109/JAS.2023.123225

37. R. R. Hossain, R. Adesunkanmi, R. Kumar, Data-driven linear Koopman embedding for networked systems: model-predictive grid control, *IEEE Syst. J.*, **17** (2023), 4809–4820. https://doi.org/10.1109/JSYST.2023.3253041

38. M. Rotulo, C. De Persis, P. Tesi, Online learning of data-driven controllers for unknown switched linear systems, *Automatica*, **145** (2022), 110519. https://doi.org/10.1016/j.automatica.2022.110519

39. C. De Persis, M. Rotulo, P. Tesi, Learning controllers from data via approximate nonlinearity cancellation, *IEEE Trans. Automat. Control*, **68** (2023), 6082–6097. https://doi.org/10.1109/TAC.2023.3234889

40. G. Z. Xie, H. Q. Jia, H. Li, Y. D. Zhong, W. L. Du, Y. Q. Dong, et al., A life prediction method of mechanical structures based on the phase field method and neural network, *Appl. Math. Model.*, **119** (2023), 782–802. https://doi.org/10.1016/j.apm.2023.03.022

41. A. Wadood, H. Albalawi, A. M. Alatwi, H. Anwar, T. Ali, Design of a novel fractional whale optimization-enhanced support vector regression (FWOA-SVR) model for accurate solar energy forecasting, *Fractal Fract.*, **9** (2025), 1–24. https://doi.org/10.3390/fractalfract9010035

AIMS Press