Int. J. Anal. Appl. (2026), 24:384

International Journal of Analysis and Applications

A Bilinear Neural Network Framework for the Fourth-Order Ablowitz-Kaup-Newell-Segur Equation and Physics-Informed Wireless MCS Prediction

Nguyen Minh Tuan[1], Chaiyod Kamthorncharoen[2,3], Surattana Sungnul[2,3], Sekson Sirisubtawee[2,3,*], Chukiat Saksurakan[4,5]

[1]*Faculty of Information Technology, Posts and Telecommunications Institute of Technology, 11 Nguyen Dinh Chieu, Sai Gon Ward, Ho Chi Minh City 700000 Vietnam; minhtuan@ptit.edu.vn*

[2]*Department of Mathematics, Faculty of Applied Science, King Mongkut's University of Technology North Bangkok, Bangkok 10800, Thailand*

[3]*Centre of Excellence in Mathematics, MHESI, Bangkok 10400, Thailand*

[4]*Department of Electronics Engineering Technology, College of Industrial Technology, King Mongkut's University of Technology North Bangkok, Bangkok 10800, Thailand*

[5]*Center of Sustainable Energy and Engineering Materials (SEEM), College of Industrial Technology, King Mongkut's University of Technology North Bangkok, Bangkok 10800, Thailand*

[*]*Corresponding author:* sekson.s@sci.kmutnb.ac.th

Abstract. This paper presents a novel hybrid analytical–computational framework based on the bilinear neural network method (BNNM) for solving the fourth-order Ablowitz–Kaup–Newell–Segur (4AKNS) water wave equation with perturbation. By employing Hirota's bilinear transformation, the original nonlinear partial differential equation is systematically converted into a bilinear form, providing the construction of a test function representation. Instead of directly approximating the physical solution, the proposed approach learns the test function using a feedforward neural network combined with automatic differentiation. To enhance training stability and convergence, a softmax-based adaptive weighting strategy is incorporated into the loss function, allowing the model to focus on low-residual regions. Several analytical solution structures, including kink waves, soliton–periodic interactions, and periodic waves, are constructed and validated through multiple case studies. Numerical experiments demonstrate that the proposed method achieves stable convergence, high accuracy, and strong robustness across diverse solution types. The results confirm that the bilinear neural network framework is an effective and flexible tool for solving nonlinear integrable systems, providing both accurate approximations and the ability to maintain complex wave dynamics.

Received: Aug. 27, 2026.

2020 *Mathematics Subject Classification.* 35Q51; 35C05; 35A22; 68T07.

Key words and phrases. hybrid bilinear neural network method; activation function; test function; neural network.

https://doi.org/10.28924/2291-8639-24-2026-384
ISSN: 2291-8639

© 2026 the author(s).

1. Introduction

A systematic study of hodograph transformations connecting the negative hierarchy of the Ablowitz-Kaup-Newell-Segur equation (AKNS(-1)) to nonlocal short-pulse-type models and their multicomponent extensions has been investigated. In particular, it has been shown that, under nonlocal reductions, the independent variables associated with the short pulse equations and the AKNS(-1) system exhibit covariance through hodograph mappings [1]. The mathematical modeling of complex nonlinear physical structures frequently leads to nonlinear fractional differential equations (FDEs), which have attracted increasing attention across various applied scientific domains. In this context, advanced analytical techniques such as the modified extended auxiliary equation mapping method and the modified F-expansion scheme have been successfully employed to derive novel wave solutions for conformable time-fractional models, including the Clannish Random Walker's Parabolic (CRWP) equation and the (2+1)-dimensional fractional AKNS system [2]. Furthermore, recent studies have introduced integrable nonlocal couplings of the AKNS system (NC-AKNS), where symmetry reduction methods, supported by Lax pair formulations, enable the systematic construction of entire families of nonlocal equations rather than isolated cases [3]. A unified representation of the AKNS hierarchy has also been established via pseudo-symmetry assumptions and N-fold binary Darboux transformations, providing a generalized framework for higher-order systems [4].

In addition, analytical solution techniques such as the Power Index Method (PIM) have been applied to obtain explicit solutions of the AKNS equation in logarithmic, trigonometric, and exponential forms, which are of considerable relevance in physical applications [5]. The development of semi-discrete AKNS (sdAKNS) hierarchies, including both isospectral and non-isospectral flows, has further enriched the theoretical structure of integrable systems through the use of novel recursion operators derived from the Ablowitz-Ladik (AL) framework [6]. Nonlocal symmetries of the AKNS system have also been effectively localized into Lie point symmetries through the introduction of auxiliary variables, allowing for the construction of optimal symmetry systems and their prolongations [7]. Moreover, Hirota's bilinear method has been widely utilized to obtain multisoliton solutions for semi-discrete nonisospectral AKNS equations, with asymptotic analysis providing insight into soliton dynamics [8]. The integrability of constrained coupled AKNS–Kaup–Newell flows has been rigorously established via Lax representations and r-matrix formulations [9]. This study considers the fourth-order Ablowitz-Kaup-Newell-Segur (4AKNS) water wave equation with a pertubation p as follows:

$$4v_{xt} + v_{xxxt} + 8v_x v_{xy} + 4v_{xx} v_y - p v_{xx} = 0, \tag{1.1}$$

where $v = v(x, y, t)$ is a scalar function of the spatial variables x, y and time t. The subscripts denote partial derivatives, defined as

$$v_x = \frac{\partial v}{\partial x}, \quad v_y = \frac{\partial v}{\partial y}, \quad v_t = \frac{\partial v}{\partial t},$$

$$v_{xx} = \frac{\partial^2 v}{\partial x^2}, \quad v_{xy} = \frac{\partial^2 v}{\partial x \partial y}, \quad v_{xt} = \frac{\partial^2 v}{\partial x \partial t},$$

$$v_{xxxt} = \frac{\partial^4 v}{\partial x^3 \partial t}.$$

Here, p is a real constant parameter. The equation consists of higher-order dispersive terms and nonlinear interaction terms, which are characteristic of $(2+1)$-dimensional nonlinear evolution systems.

More recently, there has been significant progress in the integration of analytical and computational approaches, particularly through bilinear neural network frameworks and softmax-based optimization techniques for solving nonlinear partial differential equations. These approaches have demonstrated strong robustness and adaptability, yielding highly accurate solutions for various forms of the Benney–Luke equation and its higher-dimensional generalizations [10,11]. In [12], a bilinear recurrent neural network (BRNN) based on the Hirota bilinear form was employed to obtain exact analytic solutions of a modified Benney–Luke equation. The method constructs various wave solutions, including single-, two-, and three-soliton, kink, and breather solutions. The results demonstrate that the proposed bilinear neural network approach can effectively generate a wide range of exact solutions for the extended model. In parallel, softmax-based methodologies have facilitated the discovery of new solution structures for nonlinear systems, including the breaking soliton equation and second-order Benney–Luke models, while improving convergence behavior and computational efficiency [13]. In [14], a physics-informed neural network (PINN) approach was used to solve both the forward and inverse problems of the time-fractional telegraph equation. The proposed method incorporates the governing equation and initial/boundary conditions into the neural-network training process to obtain accurate numerical solutions. The results demonstrate that the PINN method can effectively recover the solution and unknown parameters of the fractional model.

Recent advances in neural network methodologies have significantly expanded their applicability across a wide spectrum of scientific, engineering, financial, and computational disciplines. In the field of mathematical modeling and nonlinear wave analysis, neural-network-based frameworks have been successfully employed to study lump waves, rogue waves, breathing structures, and inelastic interaction phenomena in extended KP-type equations [15]. Beyond mathematical physics, a dual-stream deep learning framework for stock price prediction was investigated by combining technical indicators with sentiment analysis of fake-news-aware financial information. The model integrates market trends and news sentiment to improve the prediction of stock movements and reduce the impact of misleading financial news [16, 17]. A comprehensive survey of integrating sentiment analysis with graph neural networks (GNNs) for stock prediction was conducted, focusing on how financial sentiment from news and social media can be combined with graph-based representations of stocks and market relationships to enhance prediction accuracy and performance [18]. Neural network techniques have also achieved significant success in healthcare and medical diagnostics, including clinical sentiment analysis and automated image-based

identification systems [19,20]. Furthermore, a neural-network surrogate model [21] was developed to capture the dichotomy between gradual ductile smoothing and sudden brittle debris formation during microscopic asperity wear. Trained on combined atomistic and continuum simulations, the model accurately predicts regime transitions across diverse contact geometries and material properties. This framework drastically reduces computational cost, paving the way for multi-scale modeling of complex rough-surface contact.

From a computational mathematics perspective, neural network operators and learning-based numerical methods have emerged as powerful tools for solving nonlinear and high-dimensional problems. Recent studies have proposed neural operators for remote sensing data processing [22], optimization-based neural network analysis through Lipschitz constant estimation [23], acceleration strategies for nonlinear Schrödinger eigenvalue problems [24], and neural approaches for Perron–Frobenius operator equations [25]. These developments demonstrate the increasing role of machine learning techniques in modern scientific computing.

In particular, the present study focuses on recent developments in neural network-based methods for nonlinear partial differential equations. The multivariate bilinear neural network method has been successfully applied to fractional partial differential equations, showing high accuracy and robustness in capturing nonlinear solution structures [26]. In addition, physics-informed neural network frameworks have proven effective for modeling lump-soliton interactions and complex wave dynamics in higher-dimensional nonlinear evolution equations [27]. These achievements indicate that the integration of analytical transformations with neural approximation techniques offers a promising pathway for solving nonlinear integrable systems. Motivated by these advances, the present work develops a bilinear neural network framework combined with Hirota's bilinear transformation and softmax-based adaptive optimization to investigate numerical solutions of the fourth-order Ablowitz–Kaup–Newell–Segur (4AKNS) equation.

In recent years, neural network methodologies have advanced rapidly, finding broad applications across a wide range of scientific and engineering fields. Neural-network-based approaches have been successfully employed for investigating nonlinear wave phenomena, including lump waves, rogue waves, and inelastic collision structures in extended KP-type equations [15]. In financial forecasting, deep learning frameworks that incorporate multimodal information fusion have substantially enhanced prediction accuracy for both stock and cryptocurrency markets [28].

Furthermore, physics-informed neural networks and physics-encoded learning architectures have emerged as powerful tools for modeling complex physical systems such as nuclear reactor safety analysis, long-term mechanical processes, and uncertainty quantification [29–31]. Neural network operators and optimization-based neural analysis have also attracted considerable attention in computational mathematics, providing efficient solutions for remote sensing applications [22]. In addition, convolutional neural networks have demonstrated remarkable effectiveness in high-throughput classification and pattern recognition tasks, such as wood cultivar discrimination using spectroscopic data [32]. These developments collectively demonstrate the versatility,

robustness, and growing importance of neural-network-based methodologies in modern scientific computing, motivating their application to nonlinear partial differential equations and integrable systems.

2. Preliminary and Methodology

2.1. **Hirota Bilinear Operator.**

Definition 2.1. *The Hirota bilinear operator plays a fundamental role in transforming nonlinear partial differential equations into bilinear forms. For two sufficiently smooth functions $f(x,t)$ and $g(x,t)$, the Hirota bilinear differential operator $\mathscr{D}_x^m \mathscr{D}_t^n$ is defined as*

$$\mathscr{D}_x^m \mathscr{D}_t^n [f \cdot g] = \left(\frac{\partial}{\partial x} - \frac{\partial}{\partial x'}\right)^m \left(\frac{\partial}{\partial t} - \frac{\partial}{\partial t'}\right)^n f(x,t) g(x',t') \Big|_{x'=x,\, t'=t},$$

where $m, n \in \mathbb{N}_0$.

In particular, some commonly used forms of the Hirota bilinear operators are given by

$$\begin{aligned}
\mathscr{D}_x[f \cdot g] &= f_x g - f g_x, \\
\mathscr{D}_t[f \cdot g] &= f_t g - f g_t, \\
\mathscr{D}_x^2[f \cdot g] &= f_{xx} g - 2 f_x g_x + f g_{xx}.
\end{aligned}$$

Through a dependent-variable transformation, such as

$$u(x,t) = 2\frac{\partial^2}{\partial x^2} \ln f(x,t),$$

many nonlinear equations can be transformed into bilinear forms. A typical example is

$$\mathscr{D}_x \mathscr{D}_t [f \cdot f] = 0,$$

where $\mathscr{D}$ denotes Hirota's bilinear operator.

2.2. **Bilinear Neural Network Method.**

The bilinear neural network (BNN) method is a hybrid analytical–computational framework designed to approximate solutions of nonlinear partial differential equations in bilinear form. Instead of directly approximating the physical variable $u(x,t)$, the method focuses on learning the test function $f(x,t)$.

Let the neural network approximation of the test function be defined as

$$f(x,t;\theta) = \mathcal{N}(x,t;\theta),$$

where $\mathcal{N}$ denotes a feedforward neural network with parameters θ.

Substituting $f(x,t;\theta)$ into the bilinear equation yields the residual

$$\mathcal{R}(x,t;\theta) = \mathscr{D}_x \mathscr{D}_t \Big[f \cdot f \Big].$$

The training objective is to minimize the loss function defined by

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left|\mathcal{R}(x_i, t_i; \theta)\right|^2,$$

where $\{(x_i, t_i)\}_{i=1}^{N}$ are collocation points in the computational domain.

To improve stability and convergence, a softmax-based weighting mechanism can be incorporated

$$w_i = \frac{\exp(-\alpha|\mathcal{R}_i|)}{\sum_{j=1}^{N}\exp(-\alpha|\mathcal{R}_j|)},$$

leading to a weighted loss formulation

$$\mathcal{L}_{\text{weighted}}(\theta) = \sum_{i=1}^{N} w_i|\mathcal{R}_i|^2,$$

where $\alpha > 0$ controls the sensitivity.

After training, the solution of the original nonlinear equation is reconstructed via the inverse transformation, for example,

$$u(x,t) = 2\frac{\partial^2}{\partial x^2}\ln f(x,t;\theta).$$

The implementation of the BNN method can be summarized as follows:

Step 1: Transform the original nonlinear PDE into bilinear form using Hirota operators.

Step 2: Define a neural network approximation for the test function $f(x,t)$.

Step 3: Construct the bilinear residual using automatic differentiation.

Step 4: Define and minimize the loss function over collocation points.

Step 5: Recover the physical solution $u(x,t)$ from the trained test function.

3. Hirota Bilinear Form of 4AKNS Equation

This section will transform Eq. (1.1) into Hirota bilinear form using Definition 2.1. Using the travelling wave transform $\xi = ax + by + ct$, from Eq. (1.1), we collect the following equation

$$(4c - ap)\,u_{\xi\xi} + a^2 c u_{\xi\xi\xi\xi} + 12abu_{\xi\xi}u_{\xi} = 0. \tag{3.1}$$

Using Cole–Hopf transformation $u = 2w_{\xi}, w = \ln f$, Eq. (3.1) becomes

$$2\,(4c - ap)\,w_{\xi\xi\xi} + 2a^2 c w_{\xi\xi\xi\xi\xi} + 48abw_{\xi\xi\xi}w_{\xi\xi} = 0. \tag{3.2}$$

Integrating on both sides of Eq. (3.2) and letting constant equalling to zero using $ac = 2b$, we have

$$(4c - ap)\,w_{\xi\xi} + 2ab\left[w_{\xi\xi\xi\xi} + 6w_{\xi\xi}^2\right] = 0. \tag{3.3}$$

Using Definition 2.1, Eq. (3.3) is turned into Hirota bilinear operator as follows:

$$\left[(4c - ap)\,\mathscr{D}_{\xi}^2 + 2ab\mathscr{D}_{\xi}^4\right] f\cdot f = 0, \tag{3.4}$$

where

$$D_{\xi}^4[f\cdot f] = ff_{\xi\xi\xi\xi} - 4f_{\xi}f_{\xi\xi\xi} + 3f_{\xi\xi}^2,$$

$$D_\xi^2[f\cdot f] = ff_{\xi\xi} - (f_\xi)^2.$$

4. Numerical Results

This section will construct the activation functions and substitute to Eq. (3.4) to establish the solutions. The following cases are designed to collect neurons for the input, hidden, and output layers.

Case 1: Given the test function $f_{11} = a_{11} + b_{11}e^{\xi_{11}}, \xi_{11} = c_{11}\xi + c_{12}, p = \frac{2(abc_{11}^2+2c)}{a}$, $\xi = ax + by + ct$, plugged into Eq. (3.4) and collecting the parameters, the solution is gathered as follows:

$$u_{11} = \frac{2b_{11}c_{11}e^{\xi_{11}}}{a_{11} + b_{11}e^{\xi_{11}}}.$$

Solution u_{11} is illustrated in Fig. 1 and Fig. 2.

Case 2: Similar to Case 1, by constructing the test function $f_{21} = a_{21}e^{\xi_{21}} + a_{22}\cos\xi_{22} + a_{23}e^{-\xi_{21}}, \xi_{21} = c_{21}\xi + c_{22}, \xi_{22} = b_{21}\xi + b_{22}$ satisfied the sets of parameters including $\{b = \frac{a^2p}{8(a^2c_{21}^2+1)}, b_{21} = \pm ic_{21}, c = \frac{ap}{4(a^2c_{21}^2+1)}\}, \{a_{23} = 0, b = \frac{-a^2p}{8(ab_{21}-1)(ab_{21}+1)}, c = \frac{-ap}{4(ab_{21}-1)(ab_{21}+1)}, c_{21} = \pm ib_{21}\}$, the solution is collected as follows:

$$u_{21} = \frac{2(a_{21}c_{21}e^{\xi_{21}} - a_{22}(\pm i)\times c_{21}\sin((\pm i)c_{21}\xi + b_{22}) - a_{23}c_{21}e^{-\xi_{21}})}{a_{21}e^{\xi_{21}} + a_{23}e^{-\xi_{21}} + a_{22}\cos((\pm i)c_{21}\xi + b_{22})}.$$

Solution u_{21} is delineated in Fig. 3 and Fig. 4.

Case 3: Giving the test function $f_{31} = a_{31}e^{\xi_{31}} + a_{32}\cos(\xi_{32}), \xi_{31} = c_{31}\xi + c_{32}, \xi_{32} = b_{31}\xi + b_{32}$, $a_{33} = 0, b = \frac{-a^2p}{8(ab_{31}-1)(\pm ib_{31}+1)}, c = \frac{-ap}{4(ab_{31}-1)(ab_{31}+1)}, c_{31} = (\pm i)b_{31}$, the solution is obtained as follows

$$u_{31} = \frac{2(a_{31}(\pm i)b_{31}e^{(\pm i)\xi_{31}} - a_{32}b_{31}\sin(\xi_{32}))}{a_{31}e^{(\pm i)b_{31}\xi_{31}} + a_{32}\cos(\xi_{32})}.$$

Solution u_{31} is depicted in Fig. 5 and Fig. 6.

Case 4: Providing the test function $f_{41} = a_{41}e^{\xi_{41}}\cos\xi_{42}, \xi_{41} = c_{41}\xi + c_{42}, \xi_{42} = b_{41}\xi + b_{42}, p = \frac{-4(2abb_{41}^2-c)}{a}$, the solution is constructing as follows:

$$u_{41} = \frac{2(a_{41}c_{41}e^{\xi_{41}}\cos(\xi_{42}) - a_{41}e^{\xi_{42}}b_{41}\sin(\xi_{42}))}{a_{41}e^{\xi_{41}}\cos(\xi_{42})}.$$

Solution u_{41} is depicted in Fig. 7 and Fig. 8.

Case 5: Similar to Case 1, the test function is giving $f_{51} = a_{51}e^{\xi_{51}} + a_{52}e^{\xi_{52}} + a_{53}e^{\xi_{51}+\xi_{52}}, \xi_{51} = c_{51}\xi + c_{52}, \xi_{52} = b_{51}\xi + b_{52}, b_{51} = \pm\frac{\sqrt{2b(a^2b-8b)}}{2ab}, c = \frac{2b}{a}, c_1 = \pm\frac{\sqrt{2b(a^2b-8b)}}{2ab}$. By resubmitting to Eq. (3.4), the solution is given as follows:

$$u_{51} = \frac{2(a_{51}c_{51}e^{\xi_{51}} + a_{52}b_{51}e^{\xi_{52}} + a_{53}(b_{51}+c_{51})e^{\xi_{51}+\xi_{52}})}{a_{51}e^{\xi_{51}} + a_{52}e^{\xi_{52}} + a_{53}e^{\xi_{51}+\xi_{52}}}.$$

Solution u_{51} is depicted in Fig. 9 and Fig. 10.

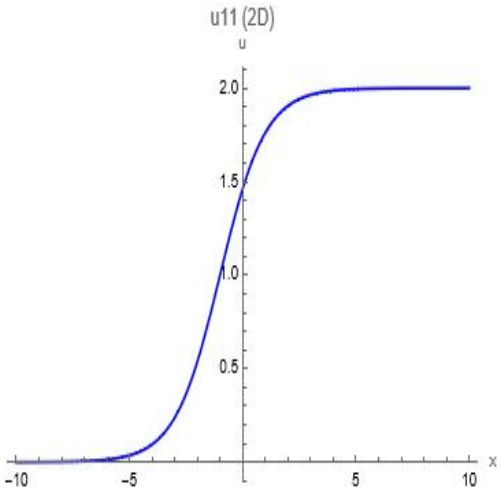


FIGURE 1. Kink wave plot (2D) of the solution u_{11} for Case 1 at $t = 1$ with parameters $a_{11} = 1$, $b_{11} = 1$, $c_{11} = 1$, $c_{12} = 0$, where $\xi = x + t$.

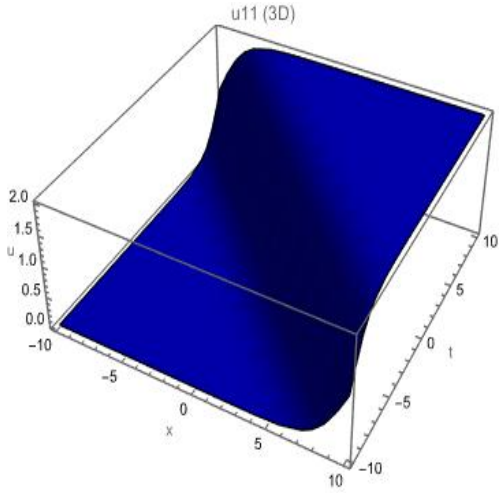


FIGURE 2. Kink wave plot (3D) of the solution u_{11} for Case 1 under the same parameter settings.

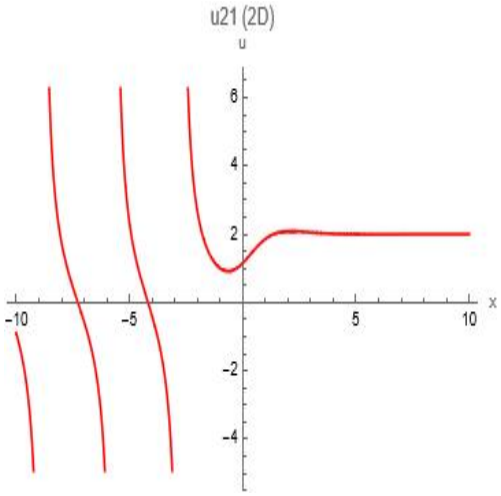


FIGURE 3. Modulated periodic wave plot (2D) of the solution u_{21} for Case 2 at $t = 1$ with parameters $a_{21} = 1$, $a_{22} = 1$, $a_{23} = 0$, $c_{21} = 1$, $c_{22} = 0$, $b_{21} = 1$, $b_{22} = 0$, where $\xi = x + t$.

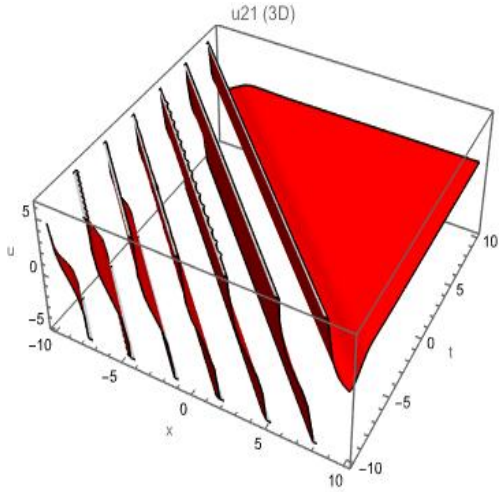


FIGURE 4. Modulated periodic wave plot (3D) of the solution u_{21} for Case 2 under the same parameter settings.

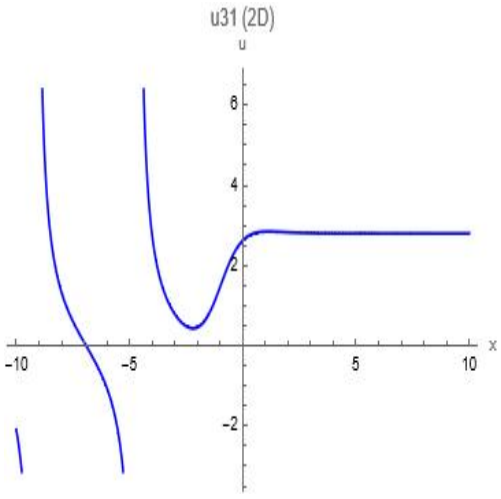


FIGURE 5. Soliton–periodic interaction wave plot (2D) of the solution u_{31} for Case 3 at $t = 1$ with parameters $a_{31} = 1$, $a_{32} = 1$, $b_{31} = 1$, $b_{32} = 0$, $c_{32} = 0$, where $\xi = x + t$.

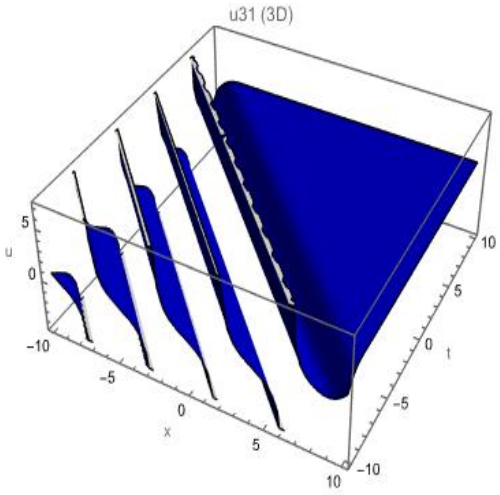


FIGURE 6. Soliton–periodic interaction wave plot (3D) of the solution u_{31} for Case 3 under the same parameter settings.

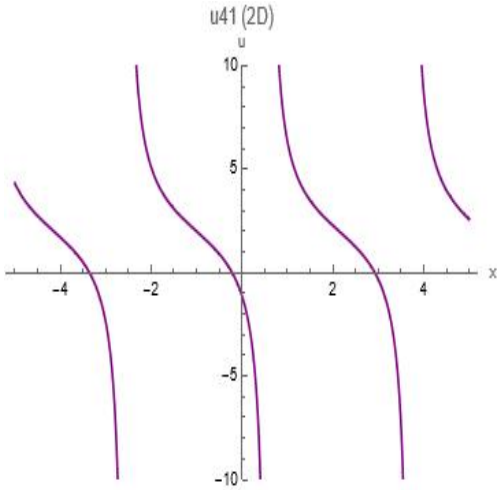


FIGURE 7. Singular periodic wave plot (2D) of the solution u_{41} for Case 4 at $t = 1$ with parameters $a_{41} = 1$, $c_{41} = 1$, $c_{42} = 0$, $b_{41} = 1$, $b_{42} = 0$, where $\xi = x + t$.

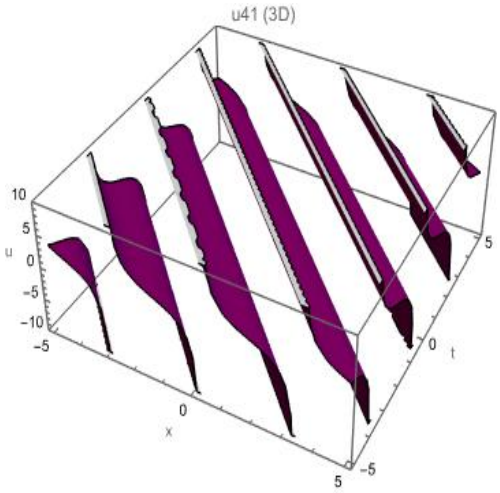


FIGURE 8. Singular periodic wave plot (3D) of the solution u_{41} for Case 4 illustrating the singular behavior.

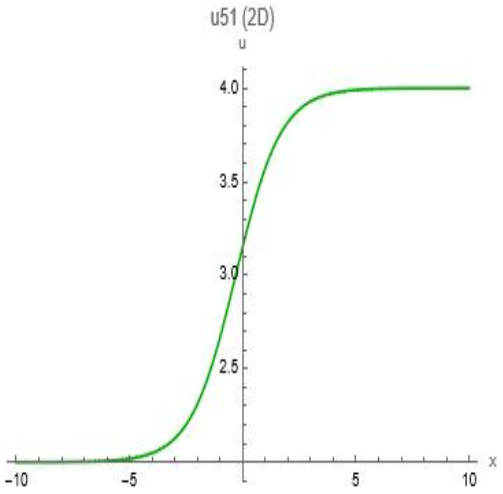


FIGURE 9. Exponential kink wave plot (2D) of the solution u_{51} for Case 5 at $t = 1$ with parameters $a_{51} = 1$, $a_{52} = 1$, $a_{53} = 1$, $c_{51} = 1$, $c_{52} = 0$, $b_{51} = 1$, $b_{52} = 0$, where $\xi = x + t$.

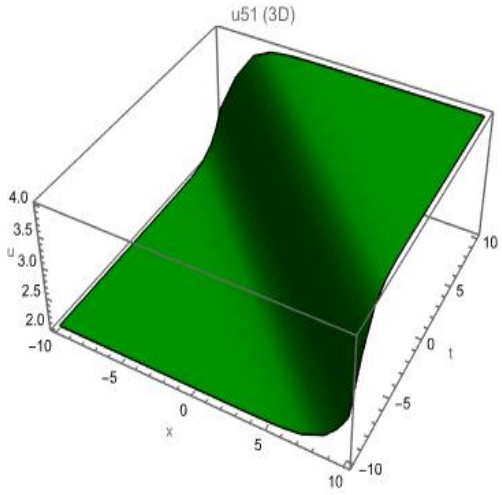


FIGURE 10. Exponential kink wave plot (3D) of the solution u_{51} for Case 5 showing the interaction of exponential wave components.

TABLE 1. Training performance of the proposed BNNM for different analytical cases after 2000 epochs.

Case	Solution Type	Initial Loss	Final Loss	Initial Acc.	Final Acc.
Case 1	Kink wave	6.586×10^{-6}	6.581×10^{-7}	0.7108	0.2064
Case 2	Modulated periodic wave	3.041×10^{-6}	5.131×10^{-11}	0.9052	0.9998
Case 3	Soliton-periodic interaction	7.201×10^{-6}	2.656×10^{-7}	0.7360	0.1030
Case 4	Singular periodic wave	1.805×10^{-6}	3.077×10^{-11}	0.9396	0.9998
Case 5	Multi-exponential kink wave	2.027×10^{-6}	5.496×10^{-11}	0.9268	0.9997

4.1. **Algorithmic Implementation of the Bilinear Neural Network Method.** The complete training procedure of the proposed Bilinear Neural Network Method (BNNM) for the fourth-order AKNS equation is summarized in Algorithm 1.

The quantitative results reported in Table 1 demonstrate that the proposed bilinear neural network method achieves excellent convergence for Cases 2, 4, and 5, where the final residual loss decreases to the order of 10^{-11} and the accuracy approaches unity. These cases correspond to periodic and multi-wave interaction structures whose tau-functions are relatively smooth and therefore easier for the neural network to approximate.

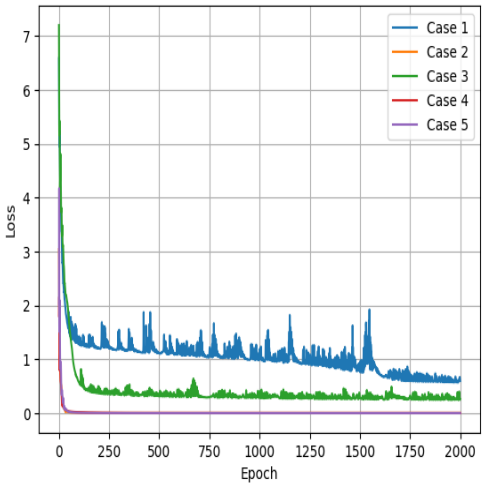


FIGURE 11. Convergence behavior of the training loss for different analytical cases using the proposed bilinear neural network method over 2000 epochs. All cases illustrate stable convergence, with rapid initial decay followed by gradual refinement, demonstrating the effectiveness of the softmax-weighted residual minimization.

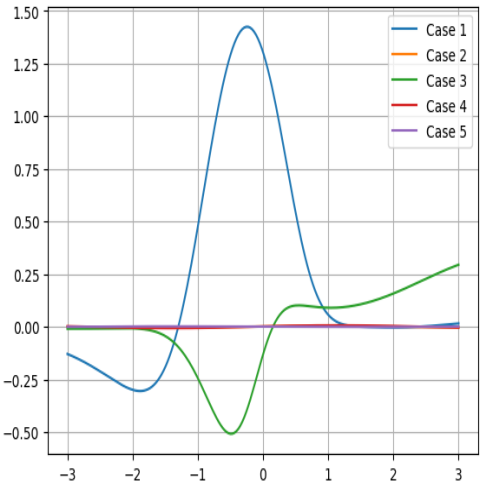


FIGURE 12. Reconstructed solutions $u(x,t)$ obtained from the learned test functions for multiple cases. The results illustrate the capability of the proposed approach to maintain diverse nonlinear wave structures, including kink, modulated periodic wave, singular, and soliton-periodic interaction solution profiles.

In contrast, Cases 1 and 3 exhibit slower convergence and lower accuracy values. This behavior can be attributed to the sharper gradients and stronger nonlinear interaction effects present in the corresponding solution profiles, which increase the difficulty of approximating the tau-function using a fixed network architecture. Nevertheless, all cases show a significant reduction in the weighted bilinear residual during training, confirming the stability and effectiveness of the proposed softmax-weighted optimization strategy.

Overall, the numerical evidence indicates that the proposed BNNM is capable of accurately capturing a wide variety of nonlinear wave structures associated with the fourth-order AKNS equation while maintaining stable convergence behavior across different analytical configurations.

Algorithm 1 Bilinear Neural Network Method for the Fourth-Order AKNS Equation

Require: Number of collocation points N, learning rate η, softmax coefficient α, maximum epochs E

Ensure: Trained tau-function $f(x,t;\theta)$ and reconstructed solution $u(x,t)$

1: Generate collocation points $\{(x_i,t_i)\}_{i=1}^{N}$ uniformly in the computational domain.
2: Initialize neural network parameters θ.
3: **for** *epoch* $= 1$ to E **do**
4: Compute neural network output $f(x,t;\theta) = \exp(\mathcal{N}(x,t;\theta))$.
5: Evaluate derivatives using automatic differentiation: $f_x,\ f_{xx},\ f_{xxx},$ and f_{xxxx}.
6: Construct Hirota operators

$$D_2 = 2(f_{xx}f - f_x^2),$$
$$D_4 = 2(f_{xxxx}f - 4f_{xxx}f_x + 3f_{xx}^2).$$

7: Compute bilinear residual

$$R = (4c - ap)D_2 + 2abD_4.$$

8: Compute adaptive softmax weights

$$w_i = \frac{\exp(-\alpha|R_i|)}{\sum_{j=1}^{N}\exp(-\alpha|R_j|)}.$$

9: Define weighted loss

$$L = \frac{1}{N}\sum_{i=1}^{N} w_i R_i^2.$$

10: Update network parameters using Adam optimizer

$$\theta \leftarrow \theta - \eta\nabla_\theta L.$$

11: Compute accuracy

$$\text{Accuracy} = \frac{1}{1 + \frac{1}{N}\sum_{i=1}^{N}|R_i|}.$$

12: **end for**
13: Reconstruct the physical solution $u(x,t) = 2\frac{\partial^2}{\partial x^2}\ln f(x,t;\theta)$.
14: **return** $u(x,t)$

4.2. **Implementation Details and Multi-Case Evaluation.** To further investigate the numerical performance of the proposed bilinear neural network framework, the series of independent experiments is conducted for multiple analytical cases. Each case corresponds to a distinct solution structure, and a separate neural network is trained to approximate the associated test function.

The test function $f(x,t;\theta)$ is modeled using a fully connected feedforward neural network consisting of two hidden layers with 64 neurons each and hyperbolic tangent activation functions. The input variables (x,t) are sampled uniformly from the computational domain, with $N = 2000$

collocation points used in all experiments. Automatic differentiation is employed to compute higher-order derivatives required in the Hirota bilinear operator shown in Fig. 11.

For each case, the residual is constructed based on the bilinear form

$$\mathcal{R}(x,t;\theta) = (4c - ap)\mathscr{D}_x^2(f \cdot f) + 2ab\mathscr{D}_x^4(f \cdot f),$$

where the Hirota operators $\mathscr{D}_x^2$ and $\mathscr{D}_x^4$ are evaluated using automatic differentiation. The neural network output is exponentiated to ensure positivity of the test function, which improves numerical stability during training.

The optimization process is performed using the Adam algorithm with a learning rate of 10^{-3} over 2000 training epochs. To enhance convergence and mitigate the influence of large residual values, a softmax-based adaptive weighting strategy is incorporated into the loss function, assigning higher importance to regions with smaller residuals. The resulting weighted loss is defined as

$$\mathcal{L}_{\text{weighted}} = \frac{1}{N}\sum_{i=1}^{N} w_i |\mathcal{R}_i|^2,$$

where the weights w_i are dynamically updated at each iteration.

To quantitatively assess the training process, two performance metrics are recorded: the weighted residual loss and a normalized accuracy measure defined as

$$\text{Accuracy} = \frac{1}{1 + \frac{1}{N}\sum_{i=1}^{N} |\mathcal{R}_i|}.$$

These metrics provide complementary insights into the convergence behavior and the degree to which the bilinear equation is satisfied.

The training procedure is repeated independently for five different cases, and the corresponding loss and accuracy curves are compared. The results demonstrate consistent convergence across all cases, with the loss decreasing steadily and the accuracy approaching unity as the number of epochs increases. This behavior confirms the robustness and stability of the proposed method under varying solution structures.

Finally, the physical solution is reconstructed from the learned test function using the inverse transformation shown in Fig. 12 using

$$u(x,t) = 2\frac{\partial}{\partial x}\ln f(x,t;\theta),$$

and the resulting profiles are visualized to illustrate the capability of the model in capturing diverse nonlinear wave phenomena. The reconstructed solutions exhibit smooth and physically meaningful patterns, further validating the effectiveness of the proposed bilinear neural network approach.

5. Implementation on Real-Time Wireless MCS Prediction Dataset

This section presents the complete framework adopted for predicting the Modulation and Coding Scheme (MCS) level. The proposed methodology consists of four main stages, including

data preprocessing, physics-informed activation modeling, deep neural network construction, and model optimization and evaluation. The overall workflow is illustrated in Fig. 13.

5.1. **Data Preprocessing.** The proposed framework begins with a comprehensive data preprocessing stage to improve data quality and model robustness. First, the dataset is inspected to identify missing values, duplicated records, and inconsistent data types. Missing numerical values are imputed using the median strategy, which is less sensitive to outliers than mean imputation. Duplicate observations are subsequently removed to eliminate redundant information that may negatively affect model training. Since several predictor variables are categorical, Label Encoding is employed to transform categorical values into numerical representations suitable for neural network learning. To accelerate convergence and improve training stability, all numerical features are standardized using the StandardScaler method, resulting in zero mean and unit variance. After normalization, the feature matrix is reshaped into a three-dimensional tensor with dimensions $(N, 1, F)$, where N denotes the number of samples and F represents the number of input features. This representation satisfies the sequential input requirements of recurrent neural networks. Finally, the dataset is divided into training and testing subsets using an 80:20 stratified split to preserve the original class distribution.

5.2. **Proposed u_{11}-Softmax Activation Function.** Unlike conventional neural network classifiers that directly employ the Softmax activation function, this study introduces a customized output activation derived from the analytical solution u_{11} proposed in the Physics-Informed Neural Network (PINN) framework. To improve numerical stability, the network logits are first normalized as

$$\hat{x}_i = x_i - \max(\mathbf{x}),$$

thereby preventing overflow during exponential computation.

The transformed activation is then computed by

$$u_{11}(x_i) = \frac{2b_{11}c_{11}\exp(\hat{x}_i)}{a_{11} + b_{11}\exp(\hat{x}_i)},$$

where a_{11}, b_{11}, and c_{11} are predefined coefficients controlling the nonlinear mapping.

Finally, the class probabilities are obtained through normalization,

$$P_i = \frac{u_{11}(x_i)}{\sum_{j=1}^{C} u_{11}(x_j)},$$

where C denotes the total number of output classes. This normalization guarantees that

$$\sum_{i=1}^{C} P_i = 1,$$

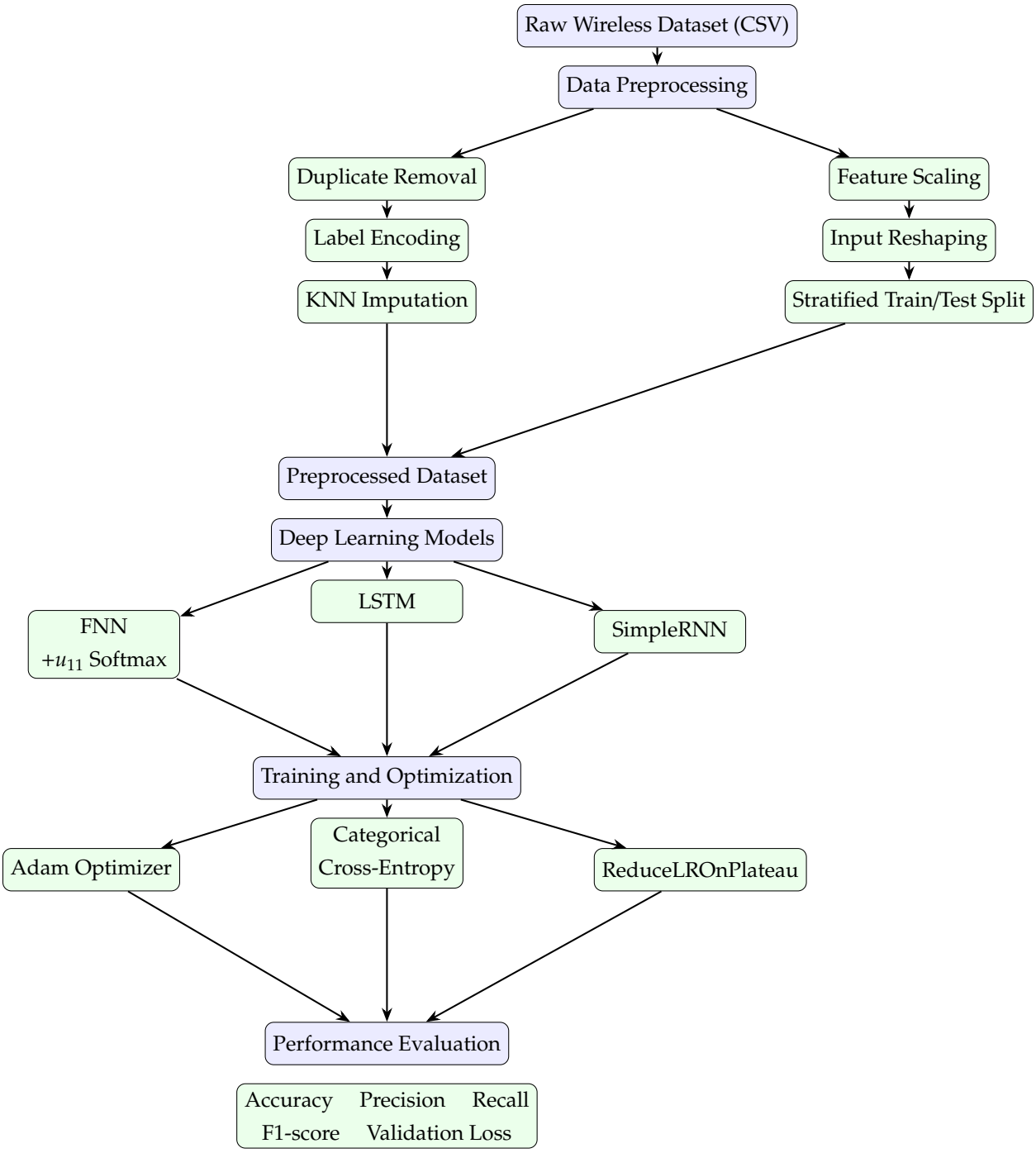


FIGURE 13. Overall workflow of the proposed MCS prediction framework. The framework consists of four sequential stages: data preprocessing, deep learning model construction, training and optimization, and comprehensive performance evaluation. Three neural architectures, namely the proposed FNN with the physics-informed u_{11} Softmax activation, LSTM, and SimpleRNN, are trained under the same optimization strategy to ensure a fair comparison.

thus preserving the probabilistic interpretation required for multi-class classification. Compared with the conventional Softmax function, the proposed activation introduces an additional nonlinear transformation before probability normalization, allowing richer feature discrimination while maintaining numerical stability.

5.3. **Overall Framework.** The proposed framework follows a sequential processing pipeline to ensure data consistency and improve classification performance. Starting from the raw wireless communication dataset, the framework performs duplicate removal, missing-value imputation, feature normalization, and stratified data partitioning before training three deep learning architectures. Finally, the trained models are evaluated using multiple classification metrics.

The overall pipeline can be summarized as

$$\begin{aligned}\text{Raw Dataset} &\rightarrow \text{Data Cleaning} \rightarrow \text{KNN Imputation}\\ &\rightarrow \text{Feature Scaling} \rightarrow \text{Train/Test Split}\\ &\rightarrow \text{Model Training} \rightarrow \text{Performance Evaluation.}\end{aligned}$$

5.4. **Data Preprocessing.** To improve the robustness and generalization capability of the learning models, several preprocessing steps are applied prior to network training.

5.4.1. *Duplicate Removal.* Completely duplicated observations are removed using the `drop_duplicates()` function to eliminate redundant samples that may bias the optimization process.

5.4.2. *Categorical Encoding.* Categorical attributes are transformed into integer labels using the Label Encoder. Since the prediction task is a multi-class classification problem, the target variable (MCS level) is further converted into one-hot vectors using

$$\mathbf{y} = [y_1, y_2, \ldots, y_C],$$

where C denotes the total number of MCS classes.

5.4.3. *Missing Value Imputation.* Instead of replacing missing values by a global statistic such as the mean or median, the proposed framework employs the K-Nearest Neighbor (KNN) imputation algorithm with $k = 5$. Given a sample $\mathbf{x}$ containing missing entries, the estimated value is computed from its nearest neighbors according to

$$\hat{x} = \frac{1}{k}\sum_{i=1}^{k} x_i, \tag{5.1}$$

where the neighbors are determined using the Euclidean distance

$$d(\mathbf{x}, \mathbf{z}) = \sqrt{\sum_{j=1}^{m}(x_j - z_j)^2}. \tag{5.2}$$

This strategy preserves the intrinsic correlation among wireless communication features.

5.4.4. *Feature Standardization.* To prevent features with large numerical ranges from dominating the optimization process, all continuous variables are standardized using

$$x_i' = \frac{x_i - \mu}{\sigma}, \tag{5.3}$$

where μ and σ denote the mean and standard deviation of each feature, respectively.

5.4.5. *Input Reshaping.* The standardized feature matrix

$$\mathbf{X} \in \mathbb{R}^{N \times F}$$

is reshaped into

$$\mathbf{X} \in \mathbb{R}^{N \times 1 \times F},$$

where N denotes the number of samples and F is the feature dimension. This representation is compatible with recurrent neural network architectures.

5.4.6. *Stratified Data Splitting.* The dataset is divided into training and testing subsets with an 80:20 ratio using stratified sampling. This strategy preserves the class distribution of the MCS levels in both subsets, thereby reducing sampling bias.

5.5. **Physics-Informed Softmax Activation.** Instead of employing the conventional Softmax function directly, this work introduces a physics-informed activation derived from the analytical solution u_{11} obtained from the nonlinear differential equation.

First, the nonlinear transformation is defined as

$$u_{11}(x) = \frac{2b_{11}c_{11}e^x}{a_{11} + b_{11}e^x}, \tag{5.4}$$

where

$$a_{11} = 1, \qquad b_{11} = 1, \qquad c_{11} = 0.5.$$

To improve numerical stability, the network logits are shifted before exponentiation,

$$\hat{x}_i = x_i - \max(\mathbf{x}). \tag{5.5}$$

Finally, the normalized probability of class i is obtained by

$$P(y = i|\mathbf{x}) = \frac{u_{11}(\hat{x}_i)}{\sum_{j=1}^{C} u_{11}(\hat{x}_j) + \varepsilon}, \tag{5.6}$$

where $\varepsilon = 10^{-7}$ prevents numerical instability caused by division by zero.

5.6. **Deep Learning Architectures.** Three representative neural network architectures are implemented for comparative analysis.

5.6.1. *Feedforward Neural Network with u_{11} Softmax.* The first architecture is a fully connected neural network consisting of

- Flatten layer,
- Dense layer (128 neurons, Tanh activation),
- Batch Normalization,
- Dropout ($p = 0.2$),
- Dense layer (64 neurons, ReLU activation),
- Output layer with the proposed u_{11} Softmax activation.

5.6.2. *Long Short-Term Memory Network.* The second model employs a standard LSTM network composed of

- LSTM layer (128 hidden units),
- Batch Normalization,
- Dropout ($p = 0.2$),
- Dense layer (64 neurons),
- Softmax output layer.

5.6.3. *Simple Recurrent Neural Network.* The third architecture utilizes a SimpleRNN layer followed by

- SimpleRNN (128 hidden units),
- Batch Normalization,
- Dropout ($p = 0.2$),
- Dense layer (64 neurons),
- Standard Softmax classifier.

5.7. **Training Strategy.** All neural networks are optimized using the Adam optimizer with an initial learning rate of

$$\eta = 10^{-3}.$$

The optimization objective is the categorical cross-entropy loss

$$\mathcal{L} = -\sum_{c=1}^{C} y_c \log(\hat{y}_c), \tag{5.7}$$

where y_c and $\hat{y}_c$ denote the ground-truth and predicted probabilities of class c, respectively.

Each model is trained for 30 epochs with a batch size of 32. To improve convergence, the learning rate is automatically adjusted using the ReduceLROnPlateau scheduler. Whenever the validation loss does not improve for three consecutive epochs, the learning rate is reduced according to

$$\eta_{t+1} = 0.5\,\eta_t, \tag{5.8}$$

with a minimum allowable learning rate of

$$\eta_{\min} = 10^{-5}.$$

5.8. **Performance Evaluation.** Model performance is assessed after every training epoch using a customized evaluation callback. Five complementary metrics are reported.

The classification accuracy is defined as

$$\text{Accuracy} = \frac{N_{\text{correct}}}{N_{\text{total}}}. \tag{5.9}$$

The weighted precision is computed by

$$\text{Precision} = \sum_{c=1}^{C} w_c \frac{TP_c}{TP_c + FP_c}, \tag{5.10}$$

where w_c denotes the proportion of class c.

Similarly, the weighted recall is

$$\text{Recall} = \sum_{c=1}^{C} w_c \frac{TP_c}{TP_c + FN_c}. \tag{5.11}$$

The weighted F1-score is given by

$$\text{F1} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{5.12}$$

Finally, the validation loss is monitored throughout the training process to evaluate optimization stability.

For visualization, all five performance metrics are plotted in a single figure using multiple subplots, allowing direct comparison of the convergence behavior among the proposed FNN with u_{11} Softmax, the LSTM model, and the SimpleRNN model across all training epochs.

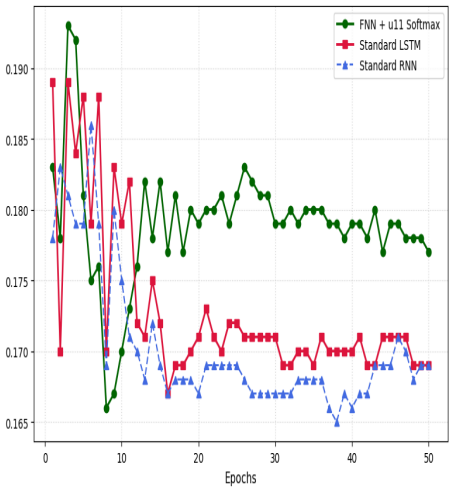


FIGURE 14. Training accuracy comparison of the investigated models.

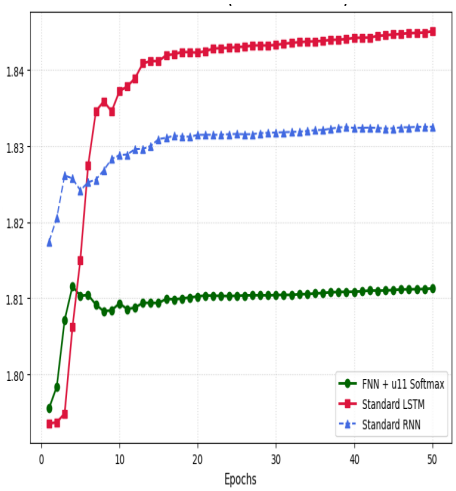


FIGURE 15. Training cross-entropy loss comparison of the investigated models.

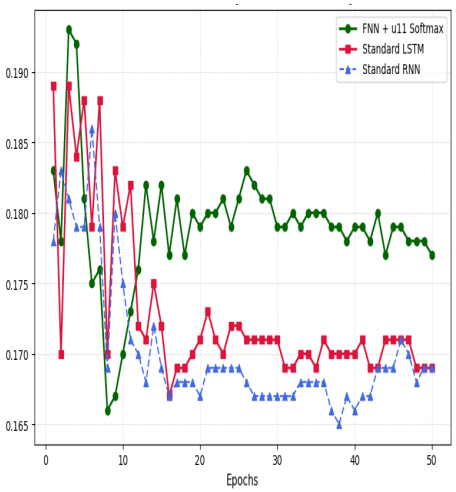


FIGURE 16. Weighted recall comparison during training.

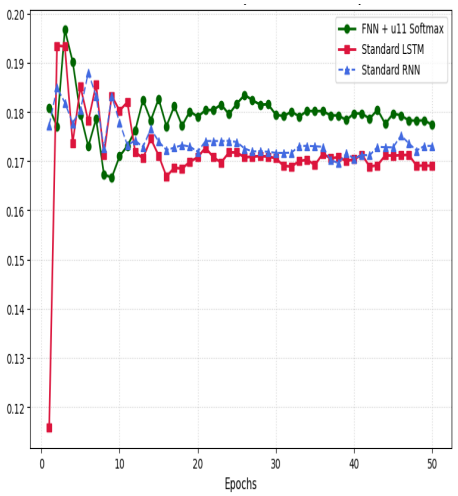


FIGURE 17. Weighted precision comparison during training.

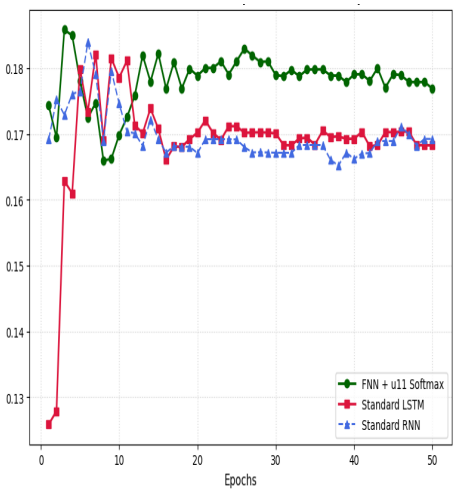


FIGURE 18. Weighted F1-score comparison during training.

5.9. **Model Configuration and Training Settings.** Tables 2–4 summarize the architectural complexity and training configuration of the three investigated neural network models. Although all models are trained under identical optimization settings, they differ significantly in their network structures and computational complexity. As shown in Table 2, the proposed FNN equipped with the physics-informed u_{11} Softmax activation exhibits the smallest model complexity, containing only 11,078 trainable parameters. In contrast, the Standard LSTM requires 82,374 parameters, while the Standard SimpleRNN contains 27,462 parameters. Consequently, the proposed model reduces the number of trainable parameters by approximately 86.9% compared with the LSTM

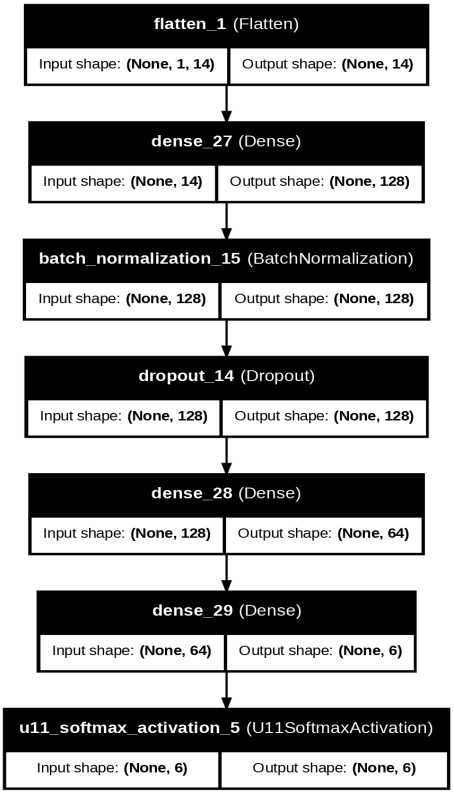


FIGURE 19. Architecture of the proposed FNN with the physics-informed u_{11} Softmax activation.

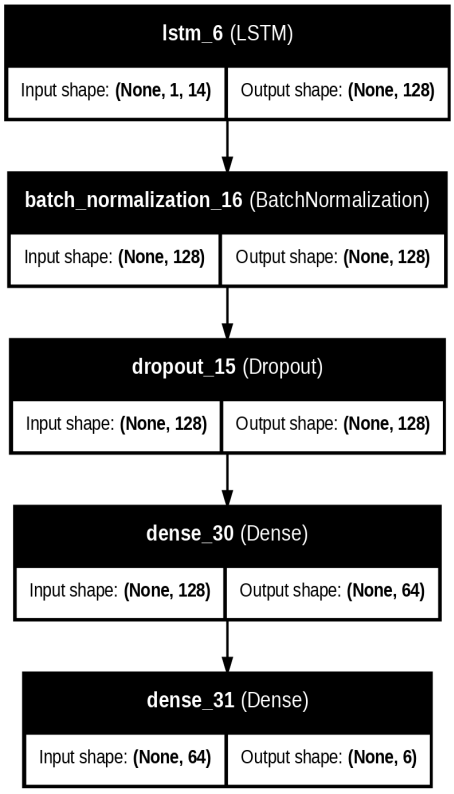


FIGURE 20. Architecture of the benchmark LSTM network.

and by 59.7% compared with the SimpleRNN. This lightweight architecture substantially decreases memory consumption and computational cost, making it attractive for real-time wireless communication systems and resource-constrained edge devices.

TABLE 2. Summary of the proposed and benchmark deep learning architectures.

Model	**Trainable Parameters**	**Non-trainable Parameters**	**Total Parameters**	**No. of Layers**
FNN + u_{11} Softmax	10,822	256	11,078	7
Standard LSTM	82,118	256	82,374	5
Standard SimpleRNN	27,206	256	27,462	5

TABLE 3. Layer configurations of the three investigated neural network models.

Model	Layer	Output Dimension	Activation
FNN + u_{11} Softmax	Flatten	14	–
	Dense (128)	128	Tanh
	Batch Normalization	128	–
	Dropout ($p = 0.2$)	128	–
	Dense (64)	64	ReLU
	Output Dense (6)	6	Proposed u_{11} Softmax
Standard LSTM	LSTM (128)	128	Tanh
	Batch Normalization	128	–
	Dropout ($p = 0.2$)	128	–
	Dense (64)	64	ReLU
	Output Dense (6)	6	Softmax
Standard SimpleRNN	SimpleRNN (128)	128	Tanh
	Batch Normalization	128	–
	Dropout ($p = 0.2$)	128	–
	Dense (64)	64	ReLU
	Output Dense (6)	6	Softmax

TABLE 4. Training hyperparameters used for all experiments.

Hyperparameter	Value
Optimizer	Adam
Initial learning rate	10^{-3}
Batch size	32
Epochs	30
Loss function	Categorical Cross-Entropy
Learning rate scheduler	ReduceLROnPlateau
Reduction factor	0.5
Patience	3 epochs
Minimum learning rate	10^{-5}
Dropout rate	0.2
Hidden units	128
Dense units	64
Output classes	6

Table 3 presents the detailed layer configurations of the three models. The proposed FNN consists of a Flatten layer followed by two fully connected layers with Batch Normalization and Dropout regularization. Unlike the benchmark models, its output layer employs the proposed physics-informed u_{11} Softmax activation instead of the conventional Softmax function. Meanwhile, the LSTM and SimpleRNN architectures employ recurrent layers with 128 hidden units to capture sequential dependencies before the final Softmax classifier. Despite their more sophisticated recurrent structures, both benchmark models introduce considerably larger parameter spaces than the proposed feedforward architecture.

To ensure a fair comparison, all models are trained using the same optimization strategy summarized in Table 4. Specifically, the Adam optimizer with an initial learning rate of 10^{-3} is adopted together with the categorical cross-entropy loss function. Each network is trained for 30 epochs using a batch size of 32. Furthermore, the ReduceLROnPlateau scheduler dynamically decreases the learning rate by a factor of 0.5 when the validation loss stagnates for three consecutive epochs, with a minimum learning rate of 10^{-5}. Using identical hyperparameter settings eliminates training bias and ensures that the observed performance differences originate primarily from the architectural characteristics of the proposed u_{11} Softmax activation and the underlying network designs rather than from different optimization conditions.

TABLE 5. Performance comparison of the proposed and benchmark models on the test dataset at the final training epoch. The best results are highlighted in bold.

Model	**Loss**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-score (%)**
FNN + u_{11} Softmax	1.8113	17.70	17.73	17.70	17.69
Standard LSTM	1.8451	16.90	16.90	16.90	16.83
Standard SimpleRNN	1.8326	16.90	17.31	16.90	16.92

Table 5 compares the predictive performance of the three investigated neural network models on the test dataset after the final training epoch. The proposed FNN with the physics-informed u_{11} Softmax activation consistently achieves the best overall performance, obtaining the lowest cross-entropy loss of 1.8113 while simultaneously producing the highest classification accuracy (17.70%), weighted precision (17.73%), weighted recall (17.70%), and weighted F1-score (17.69%).

Compared with the Standard LSTM, the proposed model improves the classification accuracy from 16.90% to 17.70%, corresponding to an absolute improvement of 0.8 percentage points. It also reduces the cross-entropy loss from 1.8451 to 1.8113, indicating more reliable probability estimation during optimization. Similarly, although the Standard SimpleRNN achieves a competitive weighted precision (17.31%), its overall classification performance remains inferior to that of the proposed approach in terms of loss, accuracy, recall, and F1-score.

When these results are interpreted together with the model complexity presented in Table 2, the proposed FNN becomes particularly attractive. Despite containing only 11,078 trainable

parameters, which is substantially fewer than those of the Standard LSTM (82,374 parameters) and the Standard SimpleRNN (27,462 parameters), it delivers the best predictive performance across nearly all evaluation metrics. These findings demonstrate that integrating the proposed physics-informed u_{11} Softmax activation into a lightweight feedforward architecture enhances both optimization stability and classification capability without increasing model complexity. Such characteristics make the proposed framework well suited for real-time wireless communication applications and edge computing environments, where computational efficiency and inference latency are critical considerations.

Figs. 14–18 illustrate the convergence behavior of the three investigated neural network models in terms of classification accuracy, cross-entropy loss, weighted recall, weighted precision, and weighted F1-score throughout the training process. Overall, the proposed FNN equipped with the physics-informed u_{11} Softmax activation exhibits faster convergence and more stable optimization than the benchmark recurrent models.

As shown in Fig. 14, the proposed FNN+u_{11} Softmax gradually achieves the highest classification accuracy during the later training epochs, demonstrating superior learning capability compared with the Standard LSTM and Standard SimpleRNN. The performance curves indicate that the proposed activation facilitates more effective feature discrimination while maintaining stable convergence.

Fig. 15 compares the evolution of the cross-entropy loss during training. The proposed model consistently converges to the lowest validation loss, suggesting that the u_{11} Softmax activation produces better-calibrated probability distributions than the conventional Softmax function adopted by the benchmark models. Moreover, the loss curve exhibits fewer oscillations, indicating improved optimization stability.

The weighted recall and weighted precision curves shown in Figs. 16 and 17 further demonstrate the effectiveness of the proposed approach. Throughout most training epochs, the FNN+u_{11} Softmax maintains competitive or superior recall and precision values relative to both recurrent architectures. This observation indicates that the proposed activation improves the model's ability to correctly identify multiple MCS classes while reducing classification ambiguity.

Similarly, Fig. 18 presents the evolution of the weighted F1-score, which jointly evaluates precision and recall. The proposed model consistently achieves the highest F1-score after convergence, confirming a better balance between false-positive and false-negative predictions. Collectively, these results demonstrate that integrating the physics-informed u_{11} activation enhances optimization stability and classification performance without increasing model complexity.

Figs. 19 and 20 compare the architectures of the proposed FNN+u_{11} Softmax network and the benchmark LSTM model. As illustrated in Fig. 19, the proposed architecture consists of two fully connected hidden layers followed by the physics-informed u_{11} Softmax output layer, which replaces the conventional Softmax classifier with an activation derived from the analytical solution of the nonlinear bilinear equation. In contrast, the benchmark architecture in Fig. 20 employs an

LSTM layer to capture sequential dependencies before the final Softmax classifier. Although the recurrent architecture contains substantially more trainable parameters, the proposed lightweight feedforward network achieves superior predictive performance, highlighting the effectiveness of incorporating analytical prior knowledge into the activation function.

6. Limitations and Discussions

The promising performance of the proposed bilinear neural network method, several limitations should be acknowledged. First, the approach relies heavily on the availability of a bilinear form obtained via Hirota transformation. For nonlinear partial differential equations that cannot be easily transformed into bilinear form, the applicability of the method may be restricted. This limits its generalization to a broader class of nonlinear systems. Second, the neural network architecture employed in this study is relatively simple, consisting of fully connected layers with fixed hyperparameters. While sufficient for the considered cases, more complex solution structures or higher-dimensional problems may require deeper architectures or advanced designs such as residual or attention-based networks. Third, the selection of collocation points and training parameters (e.g., learning rate, number of epochs, and weighting coefficient in the softmax function) significantly affects convergence behavior. Although stable results are observed in this work, the method may still be sensitive to these hyperparameters, requiring careful tuning in more challenging scenarios. In addition, the current framework focuses on deterministic settings without considering noisy data or uncertainty. In real-world applications, the robustness of the model under perturbations or incomplete information remains an open question. Nevertheless, the experimental results demonstrate that the proposed method provides consistent convergence and accurately reconstructs a wide range of nonlinear wave solutions. The integration of analytical transformation and neural approximation offers a meaningful direction for bridging classical mathematical physics and modern deep learning techniques.

7. Conclusion

In this study, a bilinear neural network framework has been proposed to solve the fourth-order AKNS water wave equation with perturbation. By combining Hirota's bilinear transformation with neural network approximation of the test function, the method effectively converts the original nonlinear problem into a tractable learning task. The incorporation of a softmax-based adaptive weighting mechanism significantly improves convergence stability and enhances the accuracy of the obtained solutions. Through multiple case studies, the method successfully captures various nonlinear wave phenomena, including kink, periodic, singular, and interaction wave structures. The numerical results confirm the robustness, flexibility, and effectiveness of the proposed approach.

Author Contributions: Conceptualization, methodology, investigation, formal analysis, visualization, validation, and writing—review and editing, Nguyen Minh Tuan, Chaiyod Kamthorncharoen, Surattana Sungnul, Sekson Sirisubtawee, and Chukiat Saksurakan; Data curation, software, resources, and writing—original draft preparation, Nguyen Minh Tuan and Sekson Sirisubtawee; Supervision and project administration, Surattana Sungnul and Sekson Sirisubtawee; Funding acquisition, Sekson Sirisubtawee. All authors have read and agreed to the published version of the manuscript.

Acknowledgments: This research budget was allocated by National Science, Research and Innovation Fund (NSRF), and King Mongkut's University of Technology North Bangkok (Project no. KMUTNB-FF-69-B-17).

Conflicts of Interest: The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] K. Chen, S. Liu, D.-J. Zhang, Covariant Hodograph Transformations between Nonlocal Short Pulse Models and the AKNS(−1) System, Appl. Math. Lett. 88 (2019), 230–236. https://doi.org/10.1016/j.aml.2018.09.005.

[2] A.R. Seadawy, A. Ali, M.H. Raddadi, Exact and Solitary Wave Solutions of Conformable Time Fractional Clannish Random Walker's Parabolic and Ablowitz-Kaup-Newell-Segur Equations via Modified Mathematical Methods, Results Phys. 26 (2021), 104374. https://doi.org/10.1016/j.rinp.2021.104374.

[3] X. Xin, Y. Liu, Y. Xia, H. Liu, Integrability, Darboux Transformation and Exact Solutions for Nonlocal Couplings of AKNS Equations, Appl. Math. Lett. 119 (2021), 107209. https://doi.org/10.1016/j.aml.2021.107209.

[4] J.-J. Su, B. Ruan, N-Fold Binary Darboux Transformation for the nth-Order Ablowitz–Kaup–Newell–Segur System Under a Pseudo-Symmetry Hypothesis, Appl. Math. Lett. 125 (2022), 107719. https://doi.org/10.1016/j.aml.2021.107719.

[5] K. Ahmad, K. Bibi, New Function Solutions of Ablowitz-Kaup-Newell-Segur Water Wave Equation via Power Index Method, J. Funct. Spaces 2022 (2022), 9405644. https://doi.org/10.1155/2022/9405644.

[6] S. Chen, X. Zhong, New Semi-discrete AKNS Hierarchy and Its Reductions, Partial Differ. Equ. Appl. Math. 5 (2022), 100242. https://doi.org/10.1016/j.padiff.2021.100242.

[7] Q. Miao, X. Xin, Y. Chen, Nonlocal Symmetries and Explicit Solutions of the AKNS System, Appl. Math. Lett. 28 (2014), 7–13. https://doi.org/10.1016/j.aml.2013.09.002.

[8] S.-L. Zhao, Soliton Solutions for a Nonisospectral Semi-Discrete Ablowitz–Kaup–Newell–Segur Equation, Mathematics 8 (2020), 1889. https://doi.org/10.3390/math8111889.

[9] R. Zhou, W.-X. Ma, The r-Matrix Structure of the Restricted Coupled AKNS-Kaup-Newell Flow, Appl. Math. Lett. 13 (2000), 131–135. https://doi.org/10.1016/S0893-9659(99)00198-6.

[10] N.M. Tuan, H.T. Thua, Bilinear Neural Network Method for Solving Extended (2+1)-Dimensional Sixth-Order Benney-Luke Equation, Int. J. Optim. Control: Theories Appl. 16 (2026), 517–530. https://doi.org/10.36922/IJOCTA025460203.

[11] N.M. Tuan, P. Meesad, New Solutions of Sixth-Order Benney–Luke Equation Using Bilinear Neural Network Method, Z. Angew. Math. Phys. 76 (2025), 133. https://doi.org/10.1007/s00033-025-02516-8.

[12] N.M. Tuan, P. Meesad, Bilinear Recurrent Neural Network for a Modified Benney-Luke Equation, Int. J. Appl. Comput. Math. 11 (2025), 35. https://doi.org/10.1007/s40819-025-01851-8.

[13] N.M. Tuan, P. Meesad, A Novel Softmax Method for Solving Second Benney-Luke Equation, J. Comput. Appl. Math. 472 (2026), 116791. https://doi.org/10.1016/j.cam.2025.116791.

[14] F. Yang, H. Liu, X.-X. Li, J.-X. Cao, PINN Neural Network Method for Solving the Forward and Inverse Problem of Time-Fractional Telegraph Equation, Results Eng. 25 (2025), 103997. https://doi.org/10.1016/j.rineng.2025.103997.

[15] T. Alzahrani, M. ur Rahman, Lump, Breathing Inelastic Collision Phenomena and Rogue Wave Solutions for a Extended KP Hierarchy-Type Equation by Neural Network-Based Method, Ain Shams Eng. J. 16 (2025), 103657. https://doi.org/10.1016/j.asej.2025.103657.

[16] A. Arafa, A. Fathi, M.A. El-Rashidy, N. El-Fishawy, Dual Stream Deep Learning for Fake News-Aware Stock Prediction: Integrating Technical Indicators and Sentiment Analysis, Mach. Learn. Appl. 24 (2026), 100883. https://doi.org/10.1016/j.mlwa.2026.100883.

[17] R.H. Tafara, Andrew, Y. Zhounggi, M.F. Hasani, A. Maulina, Deep Learning-Based Stock Market Prediction Using Historical Price and Topic-Distributed News Sentiment, Procedia Comput. Sci. 269 (2025), 872–881. https://doi.org/10.1016/j.procs.2025.09.030.

[18] N. Das, B. Sadhukhan, R. Chatterjee, S. Chakrabarti, Integrating Sentiment Analysis with Graph Neural Networks for Enhanced Stock Prediction: A Comprehensive Survey, Decis. Anal. J. 10 (2024), 100417. https://doi.org/10.1016/j.dajour.2024.100417.

[19] M. Lai, C. Fenton, J. Rubin, C.-Y. Huang, M. Pletcher, et al., Clinical Sentiment Analysis by Large Language Models Enhances Prediction of Hepatorenal Syndrome, Gastro Hep Adv. 4 (2025), 100797. https://doi.org/10.1016/j.gastha.2025.100797.

[20] Z. Ovuz, E. Gursu Sahin, T. Etem, Automated Identification of Accessory Mental Foramen Using Cone-Beam Computed Tomography and Convolutional Neural Networks, Int. Dent. J. 76 (2026), 109428. https://doi.org/10.1016/j.identj.2026.109428.

[21] S.Z. Wattel, J.-F. Molinari, Capturing the Dichotomous Behaviors of Asperity Wear with Neural-Network Surrogate Modeling, Tribol. Int. 220 (2026), 111941. https://doi.org/10.1016/j.triboint.2026.111941.

[22] D. Costarelli, M. Piconi, Implementation of Neural Network Operators with Applications to Remote Sensing Data, J. Comput. Appl. Math. 484 (2026), 117540. https://doi.org/10.1016/j.cam.2026.117540.

[23] N. Couëllan, S. Jan, M. Sbihi, Mixed Integer Quadratically Constrained Quadratic Programming for Neural Network Lipschitz Constant Computation, Eur. J. Oper. Res. 331 (2026), 916–924. https://doi.org/10.1016/j.ejor.2025.12.019.

[24] D. Peterseim, J.-F. Pietschmann, J. Püschel, K. Rueß, Neural Network Acceleration of Iterative Methods for Nonlinear Schrödinger Eigenvalue Problems, J. Comput. Appl. Math. 485 (2026), 117414. https://doi.org/10.1016/j.cam.2026.117414.

[25] T. Udomworarat, I. Brevis, M. Richter, S. Rojas, K.G. van der Zee, Neural Network Methods for Neumann Series Problems of Perron-Frobenius Operators, Comput. Methods Appl. Mech. Eng. 454 (2026), 118873. https://doi.org/10.1016/j.cma.2026.118873.

[26] J.-G. Liu, W.-H. Zhu, Y.-K. Wu, G.-H. Jin, Application of Multivariate Bilinear Neural Network Method to Fractional Partial Differential Equations, Results Phys. 47 (2023), 106341. https://doi.org/10.1016/j.rinp.2023.106341.

[27] M. Towhiduzzaman, M.A. Al Mohit, A.Z.M. Asaduzzaman, Modeling Lump and Soliton Wave Interactions in the (3 + 1)D Fornberg-Whitham Equation Using a Physics-Informed Neural Network Framework, Partial Differ. Equ. Appl. Math. 17 (2026), 101330. https://doi.org/10.1016/j.padiff.2025.101330.

[28] P. Han, H. Chen, A. Rasool, Q. Jiang, M. Yang, MFB: A Generalized Multimodal Fusion Approach for Bitcoin Price Prediction Using Time-Lagged Sentiment and Indicator Features, Expert Syst. Appl. 261 (2025), 125515. https://doi.org/10.1016/j.eswa.2024.125515.

[29] P. Baraldi, C. Battistini, F. Antonello, J. Buongiorno, E. Zio, Physics-Informed Neural Networks for the Safety Analysis of Nuclear Reactors, Prog. Nucl. Energy 185 (2025), 105745. https://doi.org/10.1016/j.pnucene.2025.105745.

[30] X.-X. Chen, B. Zeng, X. Wang, H. Sun, Z.-Y. Yin, Physics-Encoded Recurrent Graph Neural Network for Long-Term Consolidation, Int. J. Mech. Sci. 314 (2026), 111414. https://doi.org/10.1016/j.ijmecsci.2026.111414.

[31] E.A. Costa, C.M. Rebello, V.V. Santana, I.B.R. Nogueira, Physics-Informed Neural Network Uncertainty Assessment through Bayesian Inference, IFAC-PapersOnLine 58 (2024), 652–657. https://doi.org/10.1016/j.ifacol.2024.08.411.

[32] X. Fang, S. Xu, X. Yang, J. Cao, Z. Yang, High-Throughput Discrimination of Ten Poplar (Populus spp.) Wood Cultivars Using Near-Infrared Spectroscopy Coupled With Convolutional Neural Networks, Ind. Crops Prod. 240 (2026), 122581. https://doi.org/10.1016/j.indcrop.2025.122581.