UNDERSTANDING MALWARE PROPAGATION DYNAMICS THROUGH SCIENTIFIC MACHINE LEARNING

A PREPRINT

Karthik Pappu
Dakota State University
karthik.pappu@trojans.dsu.edu

Prathamesh Dinesh Joshi
Vizuara AI Labs
prathamesh@vizuara.com

Raj Abhijit Dandekar
Vizuara AI Labs
raj@vizuara.com

Rajat Dandekar
Vizuara AI Labs
rajatdandekar@vizuara.com

Sreedath Panat
Vizuara AI Labs
sreedath@vizuara.com

July 11, 2025

ABSTRACT

Accurately modeling malware propagation is essential for designing effective cybersecurity defenses, particularly against adaptive threats that evolve in real time. While traditional epidemiological models and recent neural approaches offer useful foundations, they often fail to fully capture the nonlinear feedback mechanisms present in real-world networks. In this work, we apply scientific machine learning to malware modeling by evaluating three approaches: classical Ordinary Differential Equations (ODEs), Universal Differential Equations (UDEs), and Neural ODEs. Using data from the Code Red worm outbreak, we show that the UDE approach substantially reduces prediction error compared to both traditional and neural baselines by 44%, while preserving interpretability. We introduce a symbolic recovery method that transforms the learned neural feedback into explicit mathematical expressions, revealing suppression mechanisms such as network saturation, security response, and malware variant evolution. Our results demonstrate that hybrid physics-informed models can outperform both purely analytical and purely neural approaches, offering improved predictive accuracy and deeper insight into the dynamics of malware spread. These findings support the development of early warning systems, efficient outbreak response strategies, and targeted cyber defense interventions.

Keywords Malware Propagation · Malware Dynamics · Scientific Machine Learning · Universal Differential Equations · Neural ODEs · Cybersecurity

1 Introduction

The rapid evolution of malware poses unprecedented threats to global cybersecurity infrastructure, with economic damages exceeding hundreds of billions of dollars annually [1]. Contemporary malware exhibits increasingly sophisticated behaviors, including real-time adaptation, evasion techniques, and coordinated propagation strategies that challenge traditional modeling frameworks [2, 3]. Advanced persistent threats (APTs) and zero-day malware now represent particularly challenging scenarios, employing intelligence-driven targeting and stealthy propagation mechanisms that can remain undetected for extended periods while continuing to spread [4]. These sophisticated attacks utilize decision-based targeting, where malware determines whether to actively exploit a compromised device or use it merely as a carrier for further propagation, fundamentally altering traditional infection dynamics. Understanding and predicting these complex dynamics is crucial for developing effective defense mechanisms, early warning systems, and response strategies that can adapt to emerging threats.

Traditional mathematical models of malware propagation draw inspiration from epidemiological frameworks, adapting SIR-type models originally developed for biological disease spread [5, 6] to cyber environments [7–9]. While these

approaches provide valuable baseline insights, they rely on simplified assumptions of uniform network behavior and fixed system parameters that frequently fail to capture the complex, non-linear behaviors present in real-world network environments [10, 11]. Critical phenomena such as network topology effects, traffic congestion, dynamic security countermeasures [12], and adaptive malware techniques remain challenging to model with conventional mathematical approaches, resulting in suboptimal predictions during early outbreak phases when accurate forecasting is most essential [13, 14].

The core challenge lies in the limitations of both traditional and modern approaches. Conventional epidemic models lack the flexibility to capture the complex, non-linear dynamics of modern malware spread. Meanwhile, data-driven machine learning approaches often require large amounts of labeled data (typically unavailable during emerging outbreaks) and tend to prioritize prediction accuracy at the expense of interpretability [15]. Moreover, cybersecurity data is frequently characterized by irregular sampling intervals, measurement noise, and incomplete observations, creating additional challenges for both traditional mathematical models and neural network approaches [16]. Zero-day malware detection further compounds these difficulties by providing minimal training samples for emerging threat variants, while the spatial-temporal nature of network-based propagation introduces reaction-diffusion dynamics that require sophisticated mathematical frameworks to capture effectively [17]. Although recent efforts have advanced from early theoretical models [7] to sophisticated network-based frameworks [18, 19], these still rely on fixed mathematical structures that struggle to adapt to rapidly evolving threats.

Scientific Machine Learning (SciML) offers a promising solution by integrating physics-based modeling with data-driven techniques [20]. This approach combines the interpretability of mechanistic models with the flexibility of neural networks, enabling systems to capture both fundamental dynamics and emergent behaviors that purely analytical models cannot represent [21, 22]. Recent advances in physics-informed neural networks have demonstrated success in handling irregular and noisy data across diverse domains, from vehicle platoon security [23] to automated risk assessment systems [24], suggesting their potential applicability to cybersecurity challenges.

Despite these advances, current malware models continue to face difficulties in adapting to evolving network dynamics while preserving interpretability. Critical gaps remain in handling the stochastic, spatially distributed nature of modern malware propagation, managing uncertainty in real-time operational environments, and processing irregular, noisy observational data commonly encountered in cybersecurity applications. Contemporary approaches to malware detection increasingly rely on sophisticated techniques, such as graph neural networks [25], multi-loss architectures for feature learning [26], and federated learning for distributed scenarios [27]. However, these methods often sacrifice the mechanistic insights essential for understanding propagation dynamics. It represents a critical gap in cybersecurity, where both accurate forecasting and mechanistic insight are essential.

In this paper, we make the following contributions:

1. We develop and evaluate three complementary modeling approaches (Ordinary Differential Equations (ODEs), Universal Differential Equations (UDEs), and Neural ODEs for modeling malware propagation dynamics using real-world Code Red worm data.

2. We demonstrate that the UDE approach consistently outperforms both traditional ODEs and Neural ODEs across multiple dimensions, achieving a 44% reduction in prediction error while maintaining interpretability.

3. We apply symbolic recovery to interpret the neural network component, revealing its role as a suppression mechanism that corresponds to observable cybersecurity phenomena.

4. We map recovered mathematical terms to specific cybersecurity phenomena, revealing how network saturation, security response mechanisms, and variant evolution create suppression and amplification effects that govern real-world malware propagation.

5. We provide a comprehensive analysis of each model's performance under varying conditions, including limited training data, measurement noise, and different forecasting horizons, offering practical guidance for model selection.

Our findings reveal that malware propagation in digital networks exhibits fundamentally different dynamics than traditional epidemic models suggest, with network infrastructure limitations and dynamic security responses creating natural limiting factors that conventional frameworks fail to capture [13, 28]. These insights have immediate applications for cybersecurity practitioners seeking more accurate forecasting tools and for policymakers developing evidence-based cyber defense strategies.

2 Related Work

Mathematical modeling of malware propagation has evolved from classical epidemiological frameworks to modern hybrid approaches. Foundational work by Cohen [7], and Kephart and White [8] adapted SIR models from epidemiology to digital threats. Zou et al. [9] validated these methods using empirical Code Red worm data, while later studies introduced compartmental extensions for more nuanced infection states.

Network-aware models addressed the impact of topology on outbreak dynamics. Pastor-Satorras and Vespignani [10] established that scale-free networks fundamentally alter epidemic thresholds, and Chernikova et al. [29] demonstrated these principles with real-world analysis of self-propagating malware like WannaCry using enhanced epidemiological models. Infrastructure constraints also act as natural suppressors. Staniford et al. [13] highlighted bandwidth saturation and congestion effects during Code Red, while Ganesh et al. [14] formalized bottlenecks arising from network topology, factors absent from classical epidemic models.

Despite these advances, key limitations persist. Empirical studies show that SIR-style models achieve only moderate accuracy for malware forecasting [30], while deterministic ODEs often underestimate real-world uncertainty and stochasticity [11]. Most ODE models assume homogeneous mixing, making them ill-suited for adaptive attacker behavior or dynamic defense responses. While recent extensions include agent-based models and stochastic differential equations, robust real-time forecasting and uncertainty quantification remain largely unresolved.

Modern malware poses new challenges due to zero-day attacks, polymorphism, and lateral movement techniques. Contemporary approaches have extended beyond traditional models to address these sophisticated threats. Du et al. [17] developed spatial-temporal models using partial differential equations with mixed delays. In contrast, Hernández Guillén et al. [4] introduced SCIRAS compartmental models specifically for advanced persistent threats that distinguish between carrier and targeted devices. Recent advances in graph neural networks show promise for malware propagation prediction [31]. Modern detection approaches increasingly leverage graph representation learning [25] and federated architectures [27]. While deep learning approaches have advanced static and dynamic malware detection [2], their application to propagation modeling faces challenges in balancing predictive accuracy with interpretability. Pure neural approaches often favor performance over mechanistic understanding, which can limit their effectiveness for real-time defense strategies and scenario analysis.

Scientific machine learning offers a promising integration of mechanistic and neural modeling. Neural ODEs [21] parameterize continuous-time dynamics directly with neural networks, while UDEs embed neural components within physical models for greater flexibility and interpretability [22]. Physics-informed machine learning approaches have shown particular promise for reliability and safety applications [32], though their application to cybersecurity propagation modeling remains underexplored. Symbolic regression further enhances interpretability by extracting closed-form expressions from learned neural terms [33, 34], although practical applications in cybersecurity are still in their infancy.

Deployment remains challenging due to partial observability, noisy measurements, and the use of adversarial adaptation strategies. There is also growing interest in cross-domain modeling, where methods from epidemic theory and social contagion are adapted to enhance robustness and generalization [19].

Despite these advances, no comprehensive comparison exists between classical ODEs, Neural ODEs, and UDEs for malware propagation using real-world outbreak data. Our work addresses this gap by systematically evaluating scientific machine-learning models on dynamic host-based malware data and introducing symbolic recovery techniques for interpretability. We focus specifically on outbreak dynamics, leaving static file-based detection to previous reviews [2].

3 Modeling Methodology

This section presents our data-driven modeling approach for malware dynamics. We begin with preprocessing infection data into a continuous-time representation, then progressively explore three modeling frameworks with increasing flexibility: mechanistic ordinary differential equations (ODEs), hybrid Universal Differential Equations (UDEs), and fully data-driven Neural ODEs.

3.1 Data Preprocessing Pipeline

We use the publicly available Code Red worm dataset provided by CAIDA [35], which captures darknet telescope observations of scan activity associated with the Code Red worm outbreak. Each record in the raw tab-separated dataset includes seven fields: start and end times of scanning activity (Unix timestamps), source IP addresses, top-level domain,

country, geographic coordinates (latitude/longitude), and Autonomous System (AS) metadata. These scans serve as proxies for worm infection attempts, providing temporal dynamics data suitable for mathematical modeling.

To convert this discrete event-based data into a form suitable for continuous-time modeling, we apply the following three-step preprocessing pipeline:

1. **Temporal Binning**: Raw Unix timestamps are converted to absolute datetime format and grouped into uniform 30-minute intervals using pandas date range functionality. We selected 30-minute intervals to balance temporal resolution with statistical stability, providing sufficient data points per bin while capturing the dynamic nature of worm propagation. Each bin i aggregates the number of scan events observed during that interval, yielding a discrete-time infection intensity signal I_i. Comment lines beginning with '#' in the original data are automatically filtered during this process.
2. **Smoothing**: To reduce high-frequency fluctuations in the binned intensity data and improve numerical stability for differential equation integration, we apply a 3-point moving average filter:

$$\tilde{I}_i = \begin{cases} \frac{1}{2}(I_1 + I_2) & \text{if } i = 1 \\ \frac{1}{3}(I_{i-1} + I_i + I_{i+1}) & \text{if } 1 < i < n \\ \frac{1}{2}(I_{n-1} + I_n) & \text{if } i = n \end{cases}$$

   This bounded smoothing operation preserves endpoint values while stabilizing the signal for numerical integration. The smoothing operation reduces noise while maintaining the overall temporal structure of the outbreak progression. This preprocessing approach addresses the challenges of irregular sampling and noise [16] commonly encountered in cybersecurity data.
3. **Temporal Interpolation**: A continuous-time infection intensity function $\eta(t)$ is constructed via linear interpolation of the smoothed intensity signal using scipy's interpolation methods. This enables evaluation at arbitrary time points during differential equation integration and model simulation, providing the external forcing term required for our mathematical models.

This preprocessing results in a continuous infection intensity function $\eta(t)$ suitable for integration within differential equation models.

3.2 Classical ODE Model for Malware Dynamics

We begin with a classical epidemic-style model formalized as an ordinary differential equation (ODE), designed to describe the nonlinear dynamics of malware spread in real-world networks. Grounded in foundational models of computer virus propagation [8] and complex network epidemiology [10], this formulation serves as an interpretable baseline that incorporates key features such as logistic growth, external forcing, suppression effects, and adaptive feedback mechanisms [9, 36].

To capture these dynamics in a cybersecurity context, we model the time evolution of the malware infection intensity $M(t)$ using the following nonlinear ordinary differential equation:

$$\frac{dM}{dt} = \underbrace{\alpha(t)M\left(1 - \frac{M}{K}\right)}_{\text{logistic growth}} + \underbrace{\eta(t)}_{\text{external forcing}} - \underbrace{\beta M^2}_{\text{quadratic suppression}} + \underbrace{\kappa M \log(1+M)}_{\text{adaptive feedback}} \tag{1}$$

Each term represents a specific infection mechanism:

- **Logistic growth** models intrinsic malware propagation with resource constraints, where the infection rate decays exponentially over time: $\alpha(t) = \alpha_0 \exp(-p_{\text{decay}} \cdot t/t_{\max})$
- **External forcing** incorporates the empirical infection intensity through the interpolated function $\eta(t)$
- **Quadratic suppression** captures density-dependent effects like network congestion and resource exhaustion
- **Adaptive feedback** introduces positive reinforcement representing malware propagation dynamics.

3.2.1 Parameter Specification and Optimization

Through empirical fitting to the Code Red data, we identified the following optimal parameter values:

Table 1: Optimized parameters for the malware dynamics ODE model

Parameter	Description	Value	Units
α_0	Initial infection rate	0.0501	day^{-1}
β	Suppression coefficient	10^{-4}	$\text{intensity}^{-1}\text{day}^{-1}$
κ	Feedback strength	0.005	day^{-1}
K	Carrying capacity	10^5	intensity units
p_{decay}	Temporal decay rate	0.48	dimensionless

Note that time is measured in days, and intensity units represent the number of infected hosts observed per time bin. The dimensionless parameters (p_{decay}) are normalized to ensure numerical stability during integration.

Parameters were optimized by minimizing the mean squared error between model predictions and smoothed observations:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(M(t_i;\theta) - \eta_{\text{smooth}}(t_i)\right)^2 \tag{2}$$

3.2.2 Numerical Implementation

The ODE system exhibits numerical stiffness due to the logarithmic feedback term and quadratic nonlinearity. We implement the model in Julia using the `DifferentialEquations.jl` library [37] with the following specifications:

- **Solver**: Rodas5, a 5th-order stiff-aware Rosenbrock method [38]
- **Initial condition**: $M(0) = \max(\eta(0), 1.0)$ to ensure numerical stability
- **Non-negativity constraint**: $M(t) \geq 0$ enforced at each integration step
- **Integration tolerances**: `abstol` $= 10^{-6}$, `reltol` $= 10^{-6}$

The implementation algorithm is summarized below:

Algorithm 1 Malware Dynamics ODE Implementation

function MALWARE_ODE!(du, u, p, t)
  $M \leftarrow \max(u[1], 0.0)$ ▷ Enforce non-negativity
  $\alpha, \beta, \kappa, K, p_{\text{decay}} \leftarrow p$ ▷ Unpack parameters
  $\alpha_t \leftarrow \alpha \cdot \exp(-p_{\text{decay}} \cdot t/t_{\max})$ ▷ Time-dependent rate
  growth $\leftarrow \alpha_t \cdot M \cdot (1 - M/K)$
  external $\leftarrow \eta_{\text{interp}}(t)$
  suppression $\leftarrow -\beta \cdot M^2$
  feedback $\leftarrow \kappa \cdot M \cdot \log(1 + M)$
  $du[1] \leftarrow$ growth + external + suppression + feedback
end function

3.3 Neural Ordinary Differential Equation (Neural ODE) Model

The Neural ODE approach represents a fully data-driven paradigm that replaces traditional mechanistic modeling with end-to-end learning [21]. This method directly parameterizes the entire right-hand side of the differential equation with a neural network:

$$\frac{dM}{dt} = \mathcal{N}_\psi(M, t) \tag{3}$$

where $\mathcal{N}_\psi$ is a neural network parameterized by ψ that directly learns the underlying dynamics from data without explicit mechanistic assumptions [39]. This approach offers maximum flexibility in capturing complex, nonlinear dynamics but sacrifices the interpretability provided by physics-based components [40].

3.3.1 Network Architecture

Our Neural ODE model employs a deeper architecture than the UDE counterpart to compensate for the absence of mechanistic structure. The increased capacity is necessary to learn both the fundamental dynamics and the complex interactions that the physics-based terms capture in hybrid models.

The network architecture consists of:

- **Input**: 2-dimensional vector containing normalized malware intensity $\hat{M} = M/\max(\eta)$ and normalized time $\hat{t} = t/\max(t_{\text{data}})$
- **Hidden layers**: Two dense layers with 16 neurons each using ReLU activation functions
- **Output**: Single neuron producing the time derivative dM/dt

Mathematically, the network architecture can be expressed as:

$$\mathcal{N}_\psi(M, t) = \mathbf{W}_3^T \cdot \sigma(\mathbf{W}_2 \cdot \sigma(\mathbf{W}_1 \cdot [\hat{M}, \hat{t}] + \mathbf{b}_1) + \mathbf{b}_2) + b_3 \tag{4}$$

where $\psi = \{\mathbf{W}_1 \in \mathbb{R}^{16\times 2}, \mathbf{b}_1 \in \mathbb{R}^{16}, \mathbf{W}_2 \in \mathbb{R}^{16\times 16}, \mathbf{b}_2 \in \mathbb{R}^{16}, \mathbf{W}_3 \in \mathbb{R}^{16}, b_3 \in \mathbb{R}\}$ denotes all trainable parameters and σ represents the ReLU activation function applied element-wise. The inclusion of time as an explicit input enables the network to learn time-dependent dynamics, which is critical for capturing the non-autonomous aspects of malware propagation where external factors vary over time.

3.3.2 Training and Optimization

We train the Neural ODE using a two-phase optimization strategy similar to the UDE approach, following established practices in neural differential equation optimization [21]:

1. **Adam phase**: 300 iterations using the Adam optimizer [41] with learning rate 5×10^{-4} for broad parameter exploration and robust initial convergence
2. **LBFGS phase**: 200 iterations for fine-tuning with L-BFGS, a quasi-Newton method that leverages second-order optimization for high-precision convergence

The loss function remains consistent with previous models:

$$\mathcal{L}(\psi) = \frac{1}{N}\sum_{i=1}^{N} \left(M(t_i; \psi) - \eta_{\text{smooth}}(t_i)\right)^2 \tag{5}$$

3.3.3 Implementation Details

The Neural ODE implementation incorporates several numerical techniques to ensure stability during both training and inference, addressing common challenges in neural differential equation optimization:

Algorithm 2 Neural ODE Implementation

function NEURAL_ODE!(du, u, p, t)
  $p_{\text{restructured}} \leftarrow \text{ComponentArray}(p, \text{getaxes}(p_{\text{flat}}))$ ▷ Parameter handling
  $M \leftarrow \max(\min(u[1], 5 \cdot \max_\eta), 0.0)$ ▷ State clamping for stability
  $M_{\text{norm}} \leftarrow M/\max_\eta$ ▷ Normalize intensity
  $t_{\text{norm}} \leftarrow t/\max(t_{\text{data}})$ ▷ Normalize time
  nn_input $\leftarrow \text{reshape}([M_{\text{norm}}, t_{\text{norm}}], :, 1)$ ▷ Create input tensor
  nn_out, nn_state $\leftarrow \mathcal{N}_\psi(\text{nn\_input}, p_{\text{restructured}}, \text{nn\_state})$ ▷ Forward pass
  $du[1] \leftarrow \text{clamp}(\text{nn\_out}[1], -1000.0, 1000.0)$ ▷ Output bounding
end function

Key implementation considerations include:

- **Parameter management**: ComponentArray structure for efficient gradient computation through the neural network, enabling seamless integration with automatic differentiation
- **Solver configuration**: Rodas5 stiff solver with adaptive tolerances abstol $= 10^{-3}$, reltol $= 10^{-3}$, optimized for neural differential equations

- **Numerical stability**: State clamping prevents unphysical values, input normalization ensures stable gradients, and output bounding prevents extreme derivatives that could destabilize integration
- **Error handling**: Graceful fallback mechanisms handle numerical difficulties that may arise during the iterative optimization process

This approach leverages the universal approximation capabilities of neural networks to potentially discover complex dynamics not captured by mechanistic models, albeit at the cost of reduced interpretability and increased computational requirements [42]. The method is particularly valuable when the underlying physical mechanisms are poorly understood or when the system exhibits highly nonlinear behaviors that are difficult to express analytically.

3.4 Universal Differential Equation (UDE) Model

Universal Differential Equations (UDEs) combine mechanistic models with neural networks to capture complex dynamics that are difficult to express analytically. This hybrid approach preserves interpretability while improving flexibility by embedding data-driven components into known system structures [22].

In cybersecurity applications, UDEs are particularly effective for modeling adaptive feedback mechanisms and emergent network effects that arise from complex interactions between malware propagation, network topology, and defensive responses [43]. The framework has demonstrated success across diverse scientific domains, from fluid dynamics [44] to epidemiological modeling [45], making it well-suited for malware dynamics where both mechanistic understanding and adaptive learning are essential.

Building upon our classical ODE foundation, we enhance the model by replacing the feedback term $\kappa M \log(1 + M)$ with a learnable neural network component $\mathcal{N}_\phi(M)$. This substitution yields a hybrid physics-informed model:

$$\frac{dM}{dt} = \underbrace{\alpha(t)M\left(1 - \frac{M}{K}\right)}_{\text{logistic growth}} + \underbrace{\eta(t)}_{\text{external forcing}} - \underbrace{\beta M^2}_{\text{quadratic suppression}} + \underbrace{\mathcal{N}_\phi(M)}_{\text{learned feedback}} \tag{6}$$

This formulation preserves the interpretable mechanistic components while enabling the neural network to learn complex feedback mechanisms directly from data. The approach follows the scientific machine learning principle of embedding domain knowledge while maintaining flexibility for discovery [46].

3.4.1 Neural Network Architecture

The neural network $\mathcal{N}_\phi$ is a lightweight feedforward model designed to represent unknown feedback mechanisms in the malware dynamics. It operates on normalized malware intensity values and outputs a learned correction term that complements the mechanistic structure of the differential equation.

The network architecture consists of:

- **Input layer**: A single neuron accepting the normalized malware intensity $\hat{M} = M/\max(M)$
- **Hidden layer**: One hidden layer with 10 neurons using the ReLU activation function
- **Output layer**: A single neuron that outputs the learned feedback contribution

Mathematically, the neural network can be expressed as:

$$\mathcal{N}_\phi(M) = \mathbf{W}_2^T \cdot \sigma(\mathbf{W}_1 \cdot \hat{M} + \mathbf{b}_1) + b_2 \tag{7}$$

where $\phi = \{\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2, b_2\}$ denotes the trainable parameters of the network, and σ is the ReLU activation function.

3.4.2 Training Methodology

The UDE model is trained using a two-phase optimization strategy designed to balance exploration and precision:

1. **Exploration phase**: We first perform 300 iterations using the Adam optimizer [41] with a learning rate of 5×10^{-4}. This phase enables broad exploration of the parameter space and robust initial convergence.
2. **Refinement phase**: Subsequently, we fine-tune the parameters using the L-BFGS optimizer for 200 iterations. This quasi-Newton method allows for high-precision convergence to a local minimum.

The model is trained by minimizing the mean squared error (MSE) between the predicted malware intensity trajectory and the smoothed empirical observations:

$$\mathcal{L}(\phi, \theta) = \frac{1}{N} \sum_{i=1}^{N} \left(M(t_i; \phi, \theta) - \eta_{\text{smooth}}(t_i) \right)^2 \tag{8}$$

Here, ϕ denotes the neural network parameters and θ represents the set of ODE parameters. The smoothed empirical signal $\eta_{\text{smooth}}(t)$ serves as the target trajectory for model fitting.

3.4.3 Implementation Details

Our UDE implementation incorporates several techniques to ensure stability during training:

Algorithm 3 UDE Implementation

function UDE_HYBRID!(du, u, p, t)
  $M \leftarrow \max(\min(u[1], 5 \cdot \max_\eta), 0.0)$ ▷ State clamping
  $\alpha, \beta, K, p_{\text{decay}} \leftarrow \text{abs}(p.\text{ode})$ ▷ Parameter positivity
  $\alpha_t \leftarrow \alpha \cdot \exp(-p_{\text{decay}} \cdot t / t_{\max})$
  growth $\leftarrow \alpha_t \cdot M \cdot (1 - M/K)$
  suppression $\leftarrow -\beta \cdot M^2$
  external $\leftarrow \eta_{\text{interp}}(t)$
  $M_{\text{norm}} \leftarrow M / \max_\eta$ ▷ Input normalization
  nn_input $\leftarrow$ reshape$([M_{\text{norm}}], :, 1)$ ▷ Prepare input tensor
  nn_out, nn_state $\leftarrow \mathcal{N}_\phi(\text{nn\_input}, p.\text{nn}, \text{nn\_state})$ ▷ Forward pass
  nn_term $\leftarrow$ clamp(nn_out, $-1000.0, 1000.0$) ▷ Output bounding
  $du[1] \leftarrow$ growth + suppression + external + nn_term
end function

Key implementation details include:

- **Parameter structure**: Combined ComponentArray with ODE parameters and neural network weights
- **Gradient computation**: Automatic differentiation through the ODE solver
- **Numerical stability**: Input normalization, output clamping, and parameter positivity constraints

4 Results

The evaluation emphasizes the Code Red dataset for its thorough coverage and data quality; the UDE framework aims to generalize to various malware propagation scenarios. The Code Red outbreak provides an ideal test case as it represents a well-documented, large-scale malware event with complete temporal dynamics.

We evaluate the three modeling approaches, ODE, UDE, and Neural ODE, across multiple dimensions to assess their capabilities in capturing malware dynamics.

Performance is primarily measured using Root Mean Square Error (RMSE), defined as:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \tag{9}$$

where y_i represents observed values and $\hat{y}_i$ represents model predictions.

4.1 Model Performance Comparison

Figure 1 illustrates the fitting performance of all three modeling approaches against the Code Red outbreak data. The UDE approach demonstrates improved performance (RMSE: 1281.8) compared to both the physics-based ODE model (RMSE: 2289.12) and the fully data-driven Neural ODE (RMSE: 2036.78). Table 2 provides a comprehensive comparison of performance metrics.

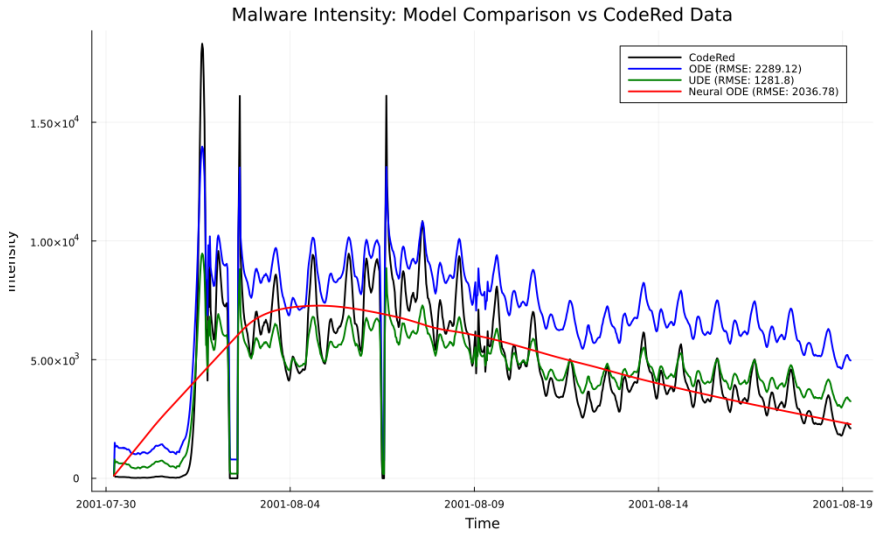

Figure 1: Comparison of model predictions against malware intensity data. The UDE model (green) achieves the lowest RMSE of 1281.8, outperforming both the traditional ODE model (blue, RMSE: 2289.12) and Neural ODE (red, RMSE: 2036.78).

Table 2: Performance metrics for malware dynamics models

Model	RMSE	MAE	MAPE (%)	Correlation
ODE	2289.12	2174.75	1503.79	0.948
Neural ODE	2036.78	1304.02	9368.04	0.687
UDE	1281.80	883.90	449.32	0.946

The UDE model achieves a 44.0% reduction in RMSE compared to the traditional ODE approach while maintaining high correlation (0.946). The pure Neural ODE model, despite its flexibility, performs worse than the hybrid approach, suggesting that completely abandoning the physics-based structure reduces model effectiveness.

4.2 Component Contribution Analysis

To understand which components drive the UDE model's performance, we conducted an ablation study by systematically evaluating different feedback mechanisms (Figure 2). This analysis isolates the contribution of each modeling component by comparing three variants:

1. **No feedback model**: ODE without any feedback term ($\kappa = 0$), RMSE: 2073.45
2. **Analytical feedback model**: ODE with standard logarithmic feedback ($\kappa M \log(1 + M)$), RMSE: 2155.8
3. **Learned feedback model**: UDE with neural network feedback, RMSE: 1281.8

The analysis reveals several key insights:

- The baseline ODE without feedback captures fundamental growth dynamics but lacks adaptive capabilities to handle complex outbreak patterns
- The standard logarithmic feedback term slightly degrades performance (2073.45 $\rightarrow$ 2155.8 RMSE), indicating that this particular analytical form may not adequately represent the true feedback mechanisms present in real malware propagation
- Replacing the fixed analytical feedback with a learnable neural network component dramatically improves performance, achieving a 40.5% reduction in RMSE compared to the analytical feedback model

This ablation study demonstrates that while physics-based structure provides valuable inductive bias for capturing fundamental dynamics, the neural network component is crucial for learning complex, non-linear feedback mechanisms that analytical forms fail to capture. The improved performance of the learned feedback approach suggests that real-world malware dynamics involve feedback processes that are more complex than simple logarithmic relationships.

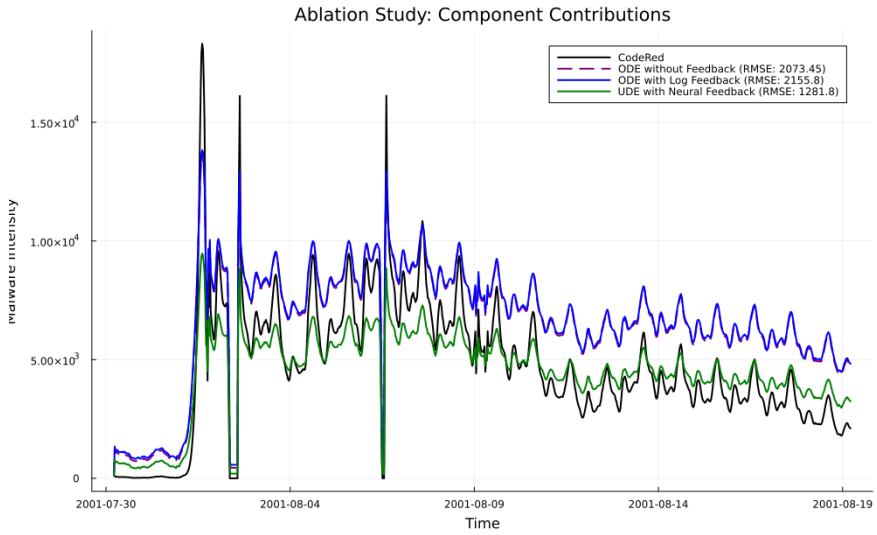

Figure 2: Ablation study comparing models with progressively more sophisticated feedback mechanisms. The UDE model with neural feedback (green, RMSE: 1281.8) substantially outperforms both the ODE without feedback (purple, RMSE: 2073.45) and ODE with standard logarithmic feedback (blue, RMSE: 2155.8).

4.3 Forecasting Capabilities

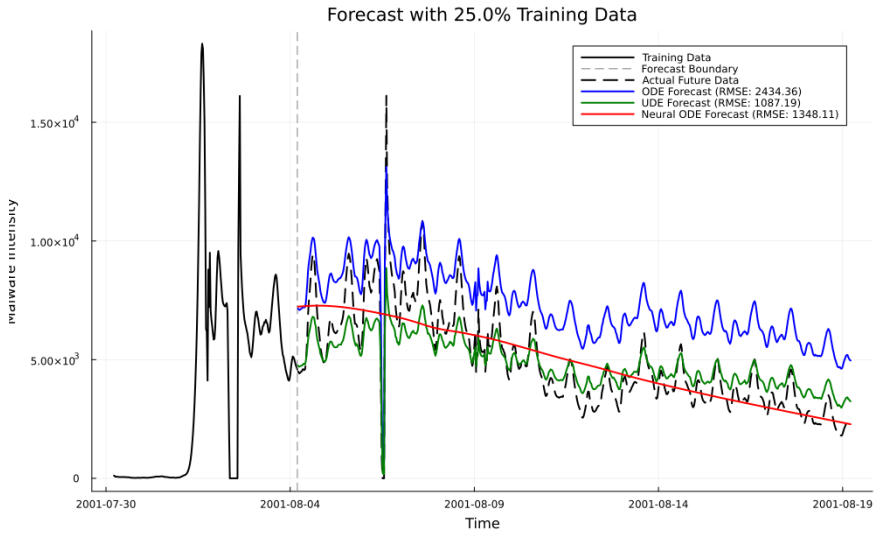

Figure 3: Forecasting performance comparison using only 25% of data for training. The UDE model (green, RMSE: 1087.19) maintains accurate predictions even with limited training data, outperforming ODE (blue, RMSE: 2434.36) and Neural ODE (red, RMSE: 1348.11).

We assessed each model's ability to forecast beyond the training data by training on different subsets of the time series. Figure 3 shows the forecast performance with just 25% training data, demonstrating UDE's remarkable ability to generalize from limited observations. The vertical dashed line indicates the boundary between training and forecasting periods.

The UDE model maintains strong forecasting performance even with minimal training data (RMSE: 1087.19 at 25%), while both the traditional ODE model (RMSE: 2434.36) and Neural ODE (RMSE: 1348.11) show significantly higher errors. Notably, the UDE model trained on only 25% of the data outperforms the Neural ODE by 19.4% and the traditional ODE by 55.3%. This enhanced forecast stability with limited data is a crucial advantage for cybersecurity applications, where early detection and response with minimal observations is essential for effective malware containment.

4.4 Robustness to Noise

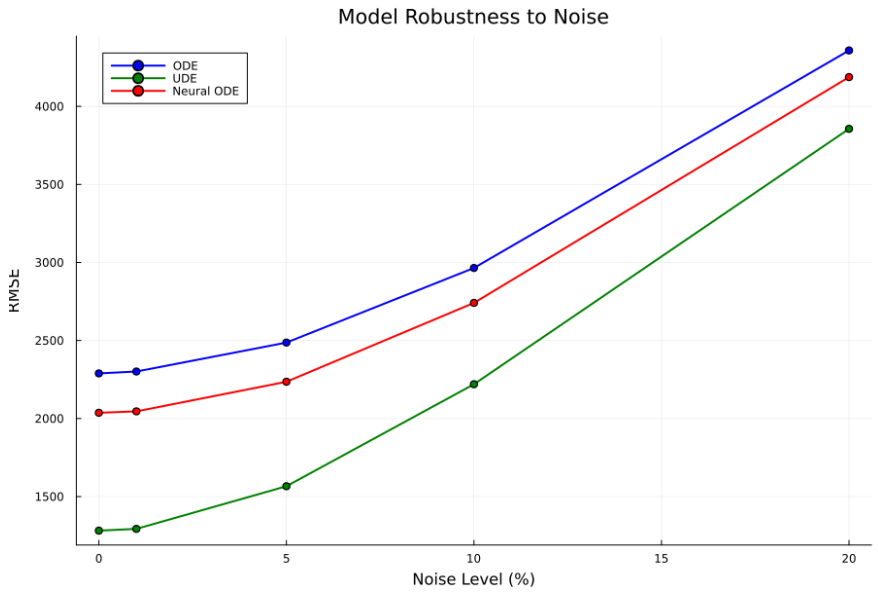

Figure 4: Model performance degradation with increasing noise levels. The UDE model (green) consistently demonstrates greater robustness to noise compared to both ODE (blue) and Neural ODE (red) approaches.

Practical malware monitoring involves substantial measurement noise due to incomplete monitoring coverage, sampling limitations, and reporting inconsistencies, making model robustness critical for real-world deployment. We evaluated each model's performance across varying noise levels (0%, 1%, 5%, 10%, and 20%), as shown in Figure 4.

The UDE model demonstrates better noise robustness across all noise levels, maintaining the lowest RMSE values throughout the tested range. At 10% noise, the UDE model (RMSE: 2219.82) substantially outperforms both the ODE (RMSE: 2964.69) and Neural ODE (RMSE: 2741.06) approaches, representing performance advantages of 25.1% and 19.0% respectively. Even at the highest tested noise level of 20%, the UDE model maintains its performance advantage over both alternatives.

This enhanced robustness likely stems from the UDE's hybrid architecture: the physics-based components provide stability against random variations by enforcing fundamental constraints, while the neural network component maintains flexibility to adapt to the underlying signal patterns despite noise contamination. This combination proves particularly valuable for cybersecurity applications where data quality can vary significantly across different monitoring infrastructures.

4.5 Symbolic Recovery and Interpretability

A key limitation of neural network approaches is their black-box nature, which hinders interpretability and limits trust in critical applications [47]. To address this challenge, we applied symbolic regression techniques to recover explicit mathematical expressions approximating the neural network's contribution (Figure 5). Following established approaches in discovering governing equations from data [33, 34], we employed regularized ridge regression [48] to identify parsimonious mathematical expressions that capture the essential dynamics while maintaining interpretability [49]. We generated candidate symbolic terms, including polynomial, logarithmic, and rational functions, and then applied ridge regression with a regularization parameter of $\lambda = 1.0$ to identify the most significant contributors while preventing overfitting. Using ridge regression with optimal regularization parameters, we identified a concise 5-term approximation that balances accuracy with interpretability:

$$\mathcal{N}(M) \approx -2608.692 \cdot \frac{M}{1+M} - 2459.9124 \cdot \log(1+M) - 2113.3055 \cdot M + 1366.5024 \cdot M^2 + 831.707 \cdot M \cdot \log(1+M) \tag{10}$$

This symbolic representation reveals several critical insights:

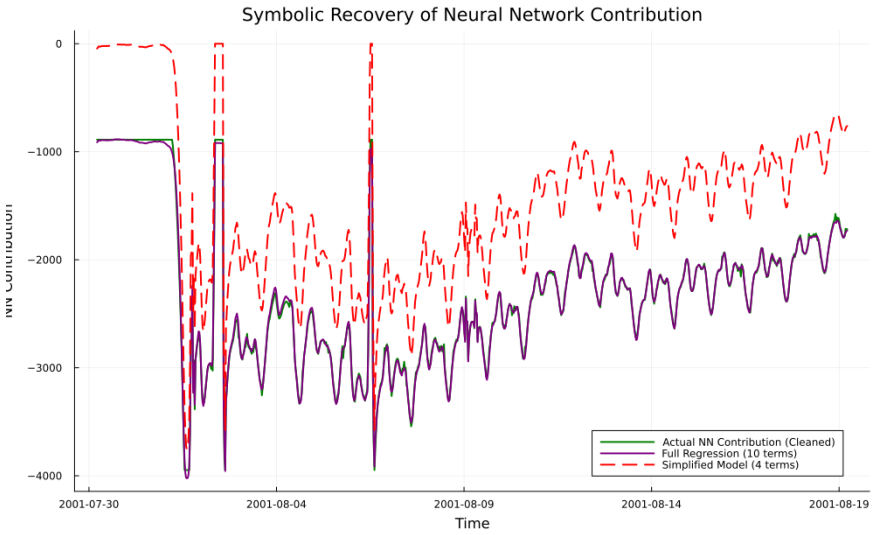

Figure 5: Symbolic recovery of the neural network contribution. The actual neural network output (green) can be approximated by a full regression model with 10 terms (purple) or a simplified model with just five terms (red dashed). The predominantly negative contribution demonstrates the neural network acting as a suppression mechanism.

1. The neural network primarily functions as a *suppression mechanism*, with an overwhelmingly negative contribution that increases with malware intensity
2. This suggests traditional epidemic models systematically overestimate malware spread in real-world networks
3. The recovered formula includes both negative (suppressive) and positive (amplifying) terms, reflecting a complex balance of competing forces in malware propagation dynamics

4.5.1 Cybersecurity Interpretation of Mathematical Terms

To translate these mathematical findings into actionable security insights, we analyzed how each term corresponds to known cybersecurity phenomena (Figure 6). This analysis bridges the gap between mathematical modeling and practical cybersecurity by mapping abstract mathematical terms to concrete network mechanisms observed in real-world malware incidents.

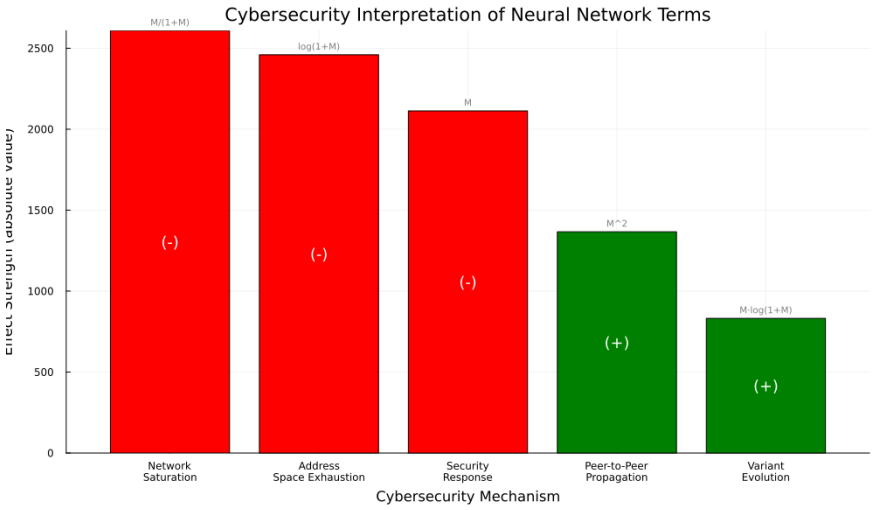

Figure 6: Cybersecurity interpretation of recovered symbolic terms. The height of each bar represents the absolute coefficient value, while the sign (+ or -) indicates whether the term accelerates or suppresses the spread of malware. The largest terms implement suppression mechanisms that correspond to real-world network effects.

Each term in Equation 10 corresponds to a specific cybersecurity mechanism:

- $\frac{M}{1+M}$ term (coefficient: -2608.692) $\rightarrow$ **Network saturation effects**, where traffic congestion and resource contention limit further spread as network infrastructure becomes overwhelmed [50].
- $\log(1+M)$ term (coefficient: -2459.9124) $\rightarrow$ **Address space exhaustion**, representing diminishing returns in finding new vulnerable hosts as larger portions of the attack surface become compromised.
- M term (coefficient: -2113.3055) $\rightarrow$ **Security response mechanisms**, including automated defenses and human-driven countermeasures deployed proportionally to observed infection levels [28].
- M^2 term (coefficient: +1366.5024) $\rightarrow$ **Peer-to-peer propagation effects**, enabling quadratic growth through direct host-to-host transmission and lateral movement within compromised networks.
- $M \cdot \log(1+M)$ term (coefficient: +831.707) $\rightarrow$ **Variant evolution and adaptation**, capturing how successful malware spawns variants and develops techniques to overcome defensive measures.

The dominant negative contributions reveal that real-world malware faces substantial limiting factors beyond those captured in traditional epidemic models. These mechanisms include network congestion, security responses, and address space constraints that actively suppress unconstrained growth. Simultaneously, the positive terms reflect how successful malware can partially overcome these limitations through adaptation and efficient propagation techniques.

This interpretation aligns with cybersecurity practitioners' understanding of real-world malware dynamics, providing actionable insights for effective defense strategies. The analysis suggests that effective countermeasures should focus on enhancing the natural suppression mechanisms, such as network segmentation to amplify saturation effects, rapid patch deployment to accelerate security responses, and threat intelligence sharing to improve collective defense capabilities.

5 Comparative Analysis of Modeling Approaches

Having evaluated each modeling approach individually across multiple dimensions, we now provide a systematic comparison to identify their relative strengths, limitations, and optimal use cases. This analysis synthesizes our experimental findings to provide practical guidance for selecting suitable modeling strategies in various cybersecurity scenarios.

Table 3: Comprehensive comparison of modeling approaches across evaluation dimensions

Evaluation Dimension	ODE	Neural ODE	UDE
Overall RMSE (Smoothed)	2289.12	2036.78	**1281.80**
Forecasting (25% training)	2434.36	1348.11	**1087.19**
Forecasting (50% training)	2696.06	912.39	**697.29**
Forecasting (75% training)	2783.45	**661.86**	777.35
Noise Robustness (10%)	2964.69	2741.06	**2219.82**
Noise Robustness (20%)	4358.34	4187.65	**3856.18**
Correlation	**0.948**	0.687	0.946
Neural Network Parameters	**0**	337	31
Interpretability	**High**	Low	**High**

5.1 Performance and Data Efficiency

The UDE approach demonstrates improved performance across most accuracy metrics, achieving a 44.0% improvement over traditional ODEs in overall fitting performance. However, analysis of forecasting capabilities reveals important data dependency patterns: while UDEs maintain consistent performance across all training data sizes (1087.19 to 777.35 RMSE), Neural ODEs show dramatic improvement with increased training data, achieving competitive results with abundant data (661.86 RMSE at 75% training) but poor performance with limited data (1348.11 RMSE at 25

This data dependency has critical implications for cybersecurity applications. UDEs trained on just 25% of data (RMSE: 1087.19) significantly outperform Neural ODEs with the same limited training data (RMSE: 1348.11), making UDEs particularly valuable for early outbreak detection scenarios where historical data is scarce. However, for forensic analysis with complete datasets, Neural ODEs can achieve competitive forecasting performance.

5.2 Robustness and Interpretability

UDEs demonstrate improved performance across all noise levels, consistently maintaining the lowest RMSE values as shown in Table 3. At high noise levels (20%), UDEs achieve approximately 8% better performance than Neural ODEs (3856.18 vs 4187.65 RMSE) and 11% better than traditional ODEs (3856.18 vs 4358.34 RMSE).

However, analysis of relative performance degradation reveals an important trade-off: while UDE maintains the best absolute performance, it experiences the highest proportional degradation from baseline (201% increase at 20% noise), compared to traditional ODE (90% increase) and Neural ODE (106% increase). This suggests that UDE's enhanced baseline performance comes with increased sensitivity to noise corruption, though it still outperforms alternatives in absolute terms across all tested noise levels.

UDEs preserve interpretability through symbolic recovery, offering an optimal balance between performance and explainability. Traditional ODEs offer full interpretability but limited performance, whereas Neural ODEs sacrifice interpretability for flexibility, as evidenced by substantially lower correlation scores (0.687) compared to both UDEs (0.946) and traditional ODEs (0.948).

5.3 Component Contribution Analysis

To understand the UDE's performance, we conducted an ablation study examining the contribution of different feedback mechanisms. This analysis reveals critical insights about the effectiveness of learned versus analytical feedback components.

Table 4: Component contribution analysis through ablation study

Model Configuration	**RMSE**	**Improvement**
ODE without Feedback	2073.45	Baseline
ODE with Log Feedback ($\kappa M \log(1 + M)$)	2155.80	-3.97%
UDE with Neural Feedback	**1281.80**	**+40.54%**

The results demonstrate that traditional analytical feedback mechanisms not only fail to improve performance but degrade it by 3.97%. This suggests that conventional epidemiological assumptions about logarithmic feedback in malware dynamics [8, 9] are inappropriate for real-world network environments where complex topology and infrastructure effects dominate [10]. In contrast, the neural network component achieves a remarkable 40.54% improvement over the baseline physics-only model, highlighting the critical importance of learned feedback mechanisms in capturing the complex dynamics of malware propagation.

This finding has profound implications: it indicates that the neural network component has discovered feedback mechanisms that are fundamentally different from traditional analytical forms, justifying the hybrid UDE approach and explaining why pure Neural ODEs, despite their flexibility, cannot match UDE performance due to the lack of physics-based structural constraints. Significantly, the UDE achieves this enhanced performance with significantly fewer trainable neural network parameters (31) compared to the Neural ODE (337 parameters), demonstrating that incorporating physics-based structure enables the neural component to focus on learning only the residual dynamics that analytical models cannot capture, rather than having to learn the entire system behavior from scratch.

5.4 Practical Recommendations and Cybersecurity Applications

Based on our comprehensive evaluation across forecasting, noise robustness, and component contribution analyses, we provide specific guidance for selecting appropriate modeling approaches in different cybersecurity scenarios:

Recommended Use Cases by Scenario:

- **Early Warning Systems (Limited Data)**: UDE is recommended due to its improved performance with minimal training data (1087.19 RMSE vs 1348.11 for Neural ODE at 25% training), making it ideal for detecting emerging threats with limited historical information [28].
- **Forensic Analysis (Complete Datasets)**: Neural ODE performs best with abundant training data (661.86 RMSE at 75% training), making it suitable for post-incident analysis where complete outbreak data is available.
- **Real-time Monitoring (Noisy Environments)**: UDE maintains the most consistent performance across all noise levels (3856.18 RMSE at 20% noise vs 4187.65 for Neural ODE), which is critical for operational environments with data quality issues.

- **Resource-Constrained Environments**: UDE offers the best balance of performance and efficiency, achieving improved accuracy with 11 times fewer neural network parameters (31 vs. 337) than Neural ODE. Traditional ODE should only be considered when neural network training is completely infeasible, accepting significant performance trade-offs.
- **Interpretable Security Analysis**: UDE provides an optimal combination of enhanced performance and interpretability through symbolic recovery, enabling security teams to understand malware spread mechanisms and inform targeted defense strategies.

UDE provides an optimal combination of enhanced performance and interpretability through symbolic recovery, enabling security teams to understand malware spread mechanisms and inform targeted defense strategies. Our findings highlight that learned feedback mechanisms capture suppression and amplification effects that traditional models often overlook. This enables accurate threat assessment, better resource allocation, and informed policy development, reducing both under-response to real threats and over-response to false alarms, thereby facilitating more efficient cyber defense planning.

6 Conclusion

This work demonstrates that Universal Differential Equations (UDEs) offer a robust framework for modeling malware propagation, combining the interpretability of physics-based models with the adaptability of neural networks. Through systematic evaluation across forecasting accuracy, noise robustness, and data efficiency, we show that hybrid physics-neural approaches consistently outperform both traditional analytical models and purely neural methods.

Our symbolic recovery analysis reveals a key insight: real-world malware propagation exhibits suppression mechanisms, such as network saturation, security response, and variant evolution, that are absent from conventional epidemiological frameworks. By translating learned neural feedback into interpretable mathematical expressions, we bridge the gap between abstract modeling and actionable cybersecurity understanding.

These findings have immediate implications for cybersecurity practice. The UDE framework's ability to perform well with limited data makes it especially suited for early warning systems during emerging outbreaks. Moreover, its interpretability empowers analysts to understand and leverage inherent network constraints for more effective intervention. The demonstrated inadequacy of traditional epidemiological feedback models suggests that malware dynamics require domain-specific formulations rather than direct analogies to biological epidemics.

While our results demonstrate the effectiveness of the UDE approach, we acknowledge some limitations in this initial study. Our evaluation relies on the Code Red worm dataset. The analysis focuses on initial outbreak dynamics and does not include statistical significance testing or confidence intervals for the reported performance improvements. Additionally, the optimal UDE parameters identified for Code Red may require retraining for different malware families, which limits their immediate applicability to emerging threats without further validation across diverse malware types.

Future work will focus on validating the UDE framework on modern malware datasets, including ransomware and botnets, to confirm the generalizability of our findings. Key priorities include developing methods to transfer UDE parameters across different malware families and testing the framework's real-time performance for early warning systems. Additionally, incorporating actual network topology data could further enhance the model's accuracy and practical applicability for cybersecurity operations.

Acknowledgments

We gratefully acknowledge the assistance of several large language models (LLMs) in preparing this manuscript. Anthropic's Claude 3.5 Sonnet was utilized for coding support with the Julia experiments and debugging assistance. Google's Gemini 2.0 Flash provided grammar correction support, while Anthropic's Claude 4 Sonnet helped resolve LaTeX formatting issues. All models were accessed via their publicly available interfaces. We thank Anthropic and Google for making these powerful tools available to support academic research.

References

[1] S. A. Afaq, M. S. Husain, A. Bello, and H. Sadia, "A critical analysis of cyber threats and their global impact," in *Computational Intelligent Security in Wireless Communications*, pp. 201–220, CRC Press, 2023.

[2] O. Or-Meir, N. Nissim, Y. Elovici, and L. Rokach, "Dynamic malware analysis in the modern era—a state of the art survey," *ACM Computing Surveys*, vol. 52, no. 5, pp. 1–48, 2019.

[3] L. Bilge and T. Dumitras, "Before we knew it: an empirical study of zero-day attacks in the real world," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, pp. 833–844, 2012.

[4] J. D. Hernández Guillén, A. Martín del Rey, and R. Casado-Vara, "Security countermeasures of a sciras model for advanced malware propagation," *IEEE Access*, vol. 7, pp. 135472–135477, 2019.

[5] W. O. Kermack and A. G. McKendrick, "A contribution to the mathematical theory of epidemics," *Proceedings of the Royal Society of London. Series A*, vol. 115, no. 772, pp. 700–721, 1927.

[6] H. W. Hethcote, "The mathematics of infectious diseases," *SIAM Review*, vol. 42, no. 4, pp. 599–653, 2000.

[7] F. Cohen, "Computer viruses: theory and experiments," *Computers & Security*, vol. 6, no. 1, pp. 22–35, 1987.

[8] J. O. Kephart and S. R. White, "Directed-graph epidemiological models of computer viruses," *Proceedings of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 343–359, 1991.

[9] C. C. Zou, W. Gong, and D. Towsley, "Code red worm propagation modeling and analysis," in *Proceedings of the 9th ACM conference on Computer and communications security*, pp. 138–147, 2002.

[10] R. Pastor-Satorras and A. Vespignani, "Dynamical and correlation properties of the internet," *Physical Review Letters*, vol. 87, no. 25, p. 258701, 2001.

[11] I. Zhuravel and S. Semenyuk, "Stochastic models for computer malware propagation," in *2024 IEEE 17th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET)*, (Lviv, Ukraine), pp. 424–427, IEEE, 2024.

[12] J. Mirkovic and P. Reiher, "A taxonomy of ddos attack and ddos defense mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[13] S. Staniford, V. Paxson, and N. Weaver, "How to own the internet in your spare time," in *Proceedings of the 11th USENIX Security Symposium*, (San Francisco, CA), pp. 149–167, USENIX Association, August 2002.

[14] A. Ganesh, L. Massoulié, and D. Towsley, "The effect of network topology on the spread of epidemics," in *Proceedings IEEE INFOCOM 2005*, vol. 2, pp. 1455–1466, 2005.

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT Press, 2016.

[16] P. Goyal and P. Benner, "Neural odes with irregular and noisy data," *arXiv preprint arXiv:2205.09489*, 2022.

[17] B. Du and H. Wang, "Partial differential equation modeling of malware propagation in social networks with mixed delays," *Computers & Mathematics with Applications*, vol. 76, no. 10, pp. 2446–2465, 2018.

[18] S. Yu, G. Gu, A. Barnawi, S. Guo, and I. Stojmenovic, "Malware propagation in large-scale networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 1, pp. 170–179, 2015.

[19] R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani, "Epidemic processes in complex networks," *Reviews of Modern Physics*, vol. 87, no. 3, pp. 925–979, 2015.

[20] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, 2021.

[21] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," *Advances in neural information processing systems*, vol. 31, 2018.

[22] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman, "Universal differential equations for scientific machine learning," *arXiv preprint arXiv:2001.04385*, 2020.

[23] S. D. Vyas, S. K. Padisala, and S. Dey, "A physics-informed neural network approach towards cyber attack detection in vehicle platoons," in *2023 American Control Conference (ACC)*, pp. 4533–4538, IEEE, 2023.

[24] Z. Wang, X. Huang, and J. Li, "Automatic cyber security risk assessment based on fuzzy fractional ordinary differential equations," *Alexandria Engineering Journal*, vol. 59, no. 6, pp. 4535–4543, 2020.

[25] T. Bilot, N. El Madhoun, K. Al Agha, and A. Zouaoui, "A survey on malware detection with graph representation learning," *ACM Computing Surveys*, vol. 56, no. 11, pp. 278:1–278:37, 2024.

[26] X. Li, Y. Chen, W. Zhang, and Q. Liu, "Multi-loss siamese neural network with batch normalization layer for malware detection," *IEEE Access*, vol. 9, pp. 69621–69632, 2021.

[27] B. S. Purkayastha, M. M. Rahman, and M. Shahpasand, "Android malware detection using machine learning and neural network: A hybrid approach with federated learning," in *2023 IEEE International Conference on Contemporary Computing and Communications (InC4)*, pp. 1–8, IEEE, 2023.

[28] C. C. Zou, L. Gao, W. Gong, and D. Towsley, "Monitoring and early warning for internet worms," in *Proceedings of the 10th ACM conference on Computer and communications security*, pp. 190–199, ACM, 2003.

[29] A. Chernikova, E. Egorova, M. I. Bocicor, and A.-J. Molnar, "Modeling self-propagating malware with epidemiological models," *Applied Network Science*, vol. 8, no. 1, pp. 1–34, 2023.

[30] E. Ginters and C. Martin-Vide, "On the usability of the sir model for malware propagation compared to diffusion models," *Computer Science and Information Systems*, vol. 20, no. 2, pp. 775–798, 2023.

[31] T. Li, Y. Liu, Q. Liu, W. Xu, Y. Xiao, and H. Liu, "A malware propagation prediction model based on representation learning and graph convolutional networks," *Digital Communications and Networks*, vol. 9, no. 5, pp. 1090–1100, 2023.

[32] Y. Xu, S. Kohtz, J. Boakye, P. Gardoni, and P. Wang, "Physics-informed machine learning for reliability and systems safety applications: State of the art and challenges," *Reliability Engineering & System Safety*, vol. 230, p. 108900, 2023.

[33] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[34] M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data," *Science*, vol. 324, no. 5923, pp. 81–85, 2009.

[35] CAIDA, "Code Red worm dataset." `https://catalog.caida.org/dataset/telescope_codered_worm`, 2001. Accessed: 2024-05-25.

[36] Z. Fang, P. Zhao, M. Xu, S. Xu, T. Hu, and X. Fang, "Statistical modeling of computer malware propagation dynamics in cyberspace," *Journal of Applied Statistics*, vol. 47, no. 5, pp. 858–883, 2020.

[37] C. Rackauckas and Q. Nie, "Differentialequations.jl – a performant and feature-rich ecosystem for solving differential equations in julia," *Journal of Open Research Software*, vol. 5, no. 1, p. 15, 2017.

[38] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, vol. 14 of *Springer Series in Computational Mathematics*. Springer Berlin, Heidelberg, 2 ed., 1996.

[39] E. Dupont, A. Doucet, and Y. W. Teh, "Augmented neural odes," in *Advances in Neural Information Processing Systems*, vol. 32, Curran Associates, Inc., 2019.

[40] P. Kidger, *On Neural Differential Equations*. Doctoral thesis, University of Oxford, 2022. arXiv preprint arXiv:2202.02435.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, (San Diego, CA), 2015. arXiv preprint arXiv:1412.6980.

[42] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[43] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[44] P. Sharma, W. T. Chung, B. Akoush, and M. Ihme, "A review of physics-informed machine learning in fluid mechanics," *Energies*, vol. 16, no. 5, p. 2343, 2023.

[45] R. Dandekar, C. Rackauckas, and G. Barbastathis, "A machine learning-aided global diagnostic and comparative tool to assess effect of quarantine control in covid-19 spread," *Patterns*, vol. 1, no. 9, p. 100145, 2020.

[46] J. D. Willard, X. Jia, S. Xu, M. S. Steinbach, and V. Kumar, "Integrating physics-based modeling with machine learning: A survey," 2020.

[47] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[48] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[49] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144, ACM, 2016.

[50] S. Staniford, J. A. Hoagland, and J. M. McAlerney, "Practical automated detection of stealthy portscans," *Journal of Computer Security*, vol. 10, no. 1-2, pp. 105–136, 2002.