Inatel
Instituto Nacional de Telecomunicações

Programa de Mestrado e Doutorado em Engenharia de Telecomunicações
TP547 – Princípios de Simulação de Sistemas de Comunicação
Trabalho 1: Obter uma figura do artigo escolhido através de simulação de Monte
Alunos: Christopher Lima & Gabriel Pivoto

Article Name: A Comparison of Monte Carlo-Based and PINN Parameter Estimation Methods for Malware Identification in IoT Networks [1].

1 Article description:

This article presents a comparative study of two advanced parameter estimation techniques, Monte Carlo-based and Physics-Informed Neural Networks (PINN), for detecting and mitigating malware in Internet of Things (IoT) networks. By analyzing real-world data and simulation scenarios, the study evaluates the effectiveness, computational efficiency, and accuracy of each method. The findings offer valuable insights into the strengths and limitations of these approaches, providing guidance for cybersecurity professionals and researchers aiming to enhance the security of IoT ecosystems against evolving malware threats.

2 Methodology

The article provides an in-depth overview of the two-parameter estimation methods under investigation: Monte Carlo-based and Physics-Informed Neural Networks (PINN). Section 2, Materials and Methods, begins by elucidating the fundamentals of Monte Carlo methods. It explains how they utilize random sampling to approximate numerical results and their application in solving complex mathematical problems. It then transitions to PINN, delineating its architecture, which integrates physics-based constraints into neural networks for enhanced accuracy and robustness. Additionally, the section outlines the key differences between the two methodologies, including their computational frameworks, data requirements, and underlying principles.

In section 3 of the article, the focus shifts to the experimental setup and methodology employed to compare the Monte Carlo-based and Physics-Informed Neural Networks (PINN) parameter estimation methods for malware identification in IoT

networks. The section outlines the specifics of the datasets used, including their sources and characteristics, to ensure a rigorous evaluation. It details the simulation scenarios and experimental protocols devised to assess the performance of each method comprehensively. Moreover, the section provides insights into the computational resources utilized, such as hardware specifications and software environments, to ensure reproducibility and transparency in the experimentation process. By meticulously detailing the experimental framework, Section 3 sets the stage for the subsequent analysis, enabling readers to understand the empirical basis for comparing the efficacy and efficiency of Monte Carlo-based and PINN methodologies in identifying malware within IoT networks.

3 Results and Conclusion

Through meticulous examination, it discusses the accuracy achieved by both approaches in detecting malware within IoT networks and compares their computational efficiency in terms of processing time and resource utilization. Furthermore, explores the robustness of Monte Carlo-based and PINN methods under diverse scenarios, encompassing varying types of malware and network complexities

The comparison of parameter estimation methods in the article reveals distinct outcomes for the three techniques assessed in the study. The Monte Carlo Mean Squared Error (MC MSE) method effectively estimates the parameters of the SIRS model but produces some estimates outside the acceptable bounds. However, the Monte Carlo Log-Square (MC log-square) model fails to estimate the parameters altogether. Conversely, the Physics-Informed Neural Network (PINN) applied to the SIRS model successfully identifies the parameters within the specified bounds. Interestingly, the PINN applied to the SIR model, as anticipated based on its training, struggles to estimate the parameters effectively. These results underscore the varying degrees of success and limitations of each method in accurately estimating parameters for the given models.

4 Monte Carlo Simulation

Upon reviewing the article, Figure 1 was deemed suitable for integration into the Monte Carlo simulation.

Given the absence of a polynomial equation or pseudo code provided in the article for generating the curve, an alternative approach was necessary for polynomial extraction. Consequently, the WebPlotDigitizer [2] platform was employed to extract the function responsible for generating the selected curve. Through this platform,

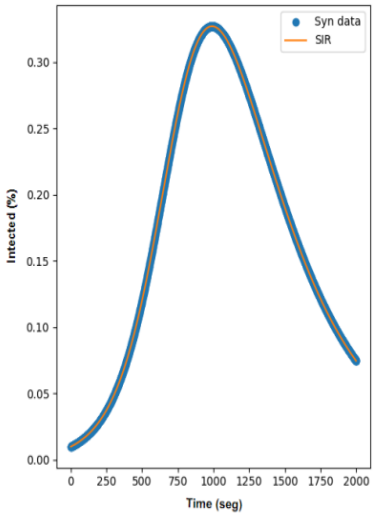

Figure 1: Chosen curve.

the image of the curve was imported, facilitating the alignment of axes for point setting. Subsequently, the curve points were extracted, enabling interpolation utilizing various polynomial types ranging from degree 1 to degree 6, alongside the option of employing a Gaussian distribution. Consequently, three distinct curve types were utilized: degree 4, degree 6, and Gaussian. Leveraging the derived functions, a Python script was developed, employing the Monte Carlo technique to insert points below the polynomial curve. Figures 2, 3 and 4 serves as a comparative analysis of the three generated graphs, showcasing the effectiveness of the approach. The implemented code is accessible on GitHub [1].

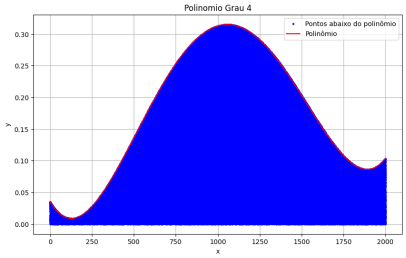

Figure 2: Polynomial approximation of degree 4.

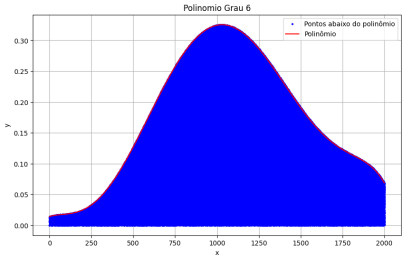

Figure 3: Polynomial approximation of degree 6.

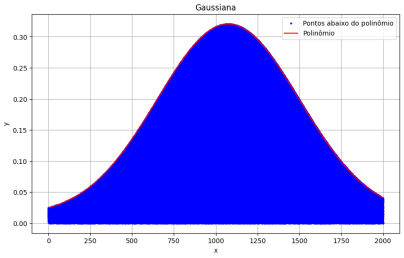

Figure 4: Gaussian approximation.

[1] https://github.com/GabrielPivoto/tp547/tree/master/SimulacaoMonteCarlo

References

[1] Marcos Severt, Roberto Casado-Vara, and Ángel Rey. "A Comparison of Monte Carlo-Based and PINN Parameter Estimation Methods for Malware Identification in IoT Networks". In: *Technologies* 11 (Sept. 2023), p. 133. DOI: 10.3390/technologies11050133.

[2] Ankit Rohatgi. *WebPlotDigitizer*. URL: https://automeris.io/WebPlotDigitizer.html.