

Performance Analysis of Different ML/DL algorithms for Detecting Web Attacks

Dendukuri Lakshmi Harshitha

PG scholar, Department of MCA, CDNR collage, Bhimavaram, Andhra Pradesh.

V.Sarala

(Assistant Professor), Master of Computer Applications, DNR collage, Bhimavaram, Andhra Pradesh.

Abstract

In the digital era, where web applications serve as the backbone for numerous services, the security of online systems has become a critical concern. Web attacks such as SQL injection, cross-site scripting (XSS), denial-of-service (DoS), and phishing have become increasingly sophisticated, threatening the integrity, confidentiality, and availability of web-based platforms. Detecting these threats promptly is crucial to minimizing damage and securing data.

Traditional security mechanisms like firewalls and rule-based systems often struggle to detect novel or obfuscated attacks, prompting the need for more adaptive and intelligent detection techniques. Machine Learning (ML) and Deep Learning (DL) models have emerged as promising tools for identifying patterns and anomalies within network traffic and web application logs.

This project aims to perform a comparative analysis of various ML and DL algorithms, such as Support Vector Machine (SVM), Decision Trees, Random Forests, Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN), to evaluate their performance in detecting different types of web attacks.

1. The goal is to assess the accuracy, precision, recall, F1-score, and overall robustness of each model using benchmark datasets. The study intends to provide insight into which algorithms are most effective under specific attack conditions, helping cybersecurity professionals choose appropriate detection mechanisms.

Introduction

1. With the explosive growth of the internet and web-based technologies, cyber threats have evolved, targeting critical

infrastructure and sensitive user data. The frequency and complexity of web attacks

2. have made conventional detection methods insufficient, leading to severe financial and reputational losses for organizations.
3. Web attacks typically exploit vulnerabilities in web applications or networks, and may include injection attacks, session hijacking, cross-site scripting, and others. These attacks can lead to unauthorized access, data leakage, or system disruptions, necessitating advanced detection systems.
4. Machine Learning and Deep Learning approaches offer dynamic, data-driven solutions for identifying abnormal patterns indicative of malicious activity. These models learn from historical data and adapt to detect new forms of attacks without relying solely on predefined rules.
5. This research focuses on analyzing the performance of different ML and DL algorithms in detecting web attacks. The objective is to determine which algorithms are more effective and efficient in various attack scenarios, thereby improving the overall security posture of web applications.

Literature Survey

1. Various studies have demonstrated the effectiveness of ML in intrusion detection. For example, SVM and Random Forest classifiers have been widely used for their robustness and high accuracy in binary and multi-class classification problems related to web attacks.
2. Deep learning models, especially CNNs and RNNs, have shown significant

improvements over traditional ML methods by automatically extracting features from raw data and capturing temporal relationships in network traffic.

3. Research by Shone et al. (2018) proposed a deep autoencoder-based approach for intrusion detection, achieving better detection rates than classical models. Similarly, studies using LSTM networks have reported strong performance in detecting time-based attack sequences.
4. However, these models vary in their computational complexity and suitability for real-time deployment. Thus, comparative studies are essential to understand the trade-offs between detection performance, training time, and resource consumption.

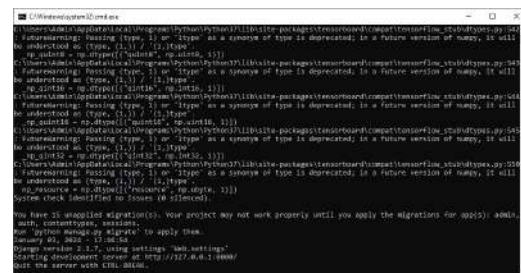
Existing Method

1. Traditional intrusion detection systems (IDS) primarily rely on signature-based or rule-based methods, which require predefined rules to detect known attack patterns. While efficient for known threats, they fail to detect zero-day or slightly modified attacks.
2. ML-based IDS typically involve data preprocessing, feature extraction, and classification. Models like Decision Trees, Naïve Bayes, and K-Nearest Neighbors have been implemented to classify network traffic or HTTP requests into benign or malicious categories.
3. These methods offer higher detection rates than rule-based systems but often require manual feature engineering, which can be both time-consuming and error-prone. Additionally, they may not adapt well to evolving attack patterns.
4. Despite their limitations, existing ML-based approaches form a strong baseline for evaluating the performance of more advanced DL models. Understanding their strengths and weaknesses helps in designing improved detection frameworks.

1. This project proposes a comparative evaluation of ML and DL models, focusing on their effectiveness in detecting various web attacks using publicly available datasets such as CICIDS2017, NSL-KDD, and UNSW-NB15.
2. The ML models considered include SVM, Random Forest, and Logistic Regression, while DL models include CNN, LSTM, and hybrid CNN-LSTM architectures. Each model will be trained, validated, and tested using the same data splits for fair comparison.
3. Preprocessing steps will involve normalization, feature selection, and handling imbalanced data using techniques like SMOTE. Performance metrics such as accuracy, recall, precision, F1-score, and AUC will be used to assess model effectiveness.
4. The final goal is to determine which algorithms offer the best balance between detection accuracy and computational efficiency, and to provide a scalable and practical detection solution for web-based systems.

reluse

To run web code double click on 'run.bat' file to start python DJANGO server and will get below screen



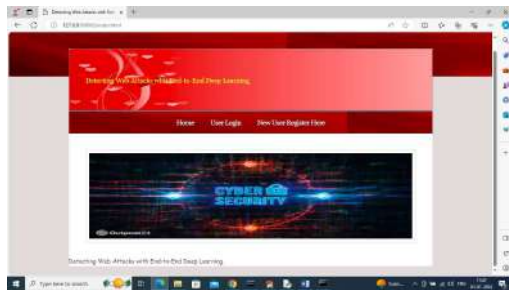
```

C:\WebSecurity>python manage.py runserver 0.0.0.0:8000
Watching for file changes with Statistic
Performing system checks...
System check identified no issues (0 silenced)
You have 15 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Now system ready to migrate. To apply them, run 'python manage.py migrate'
Starting development server at http://0.0.0.0:8000/
Quit the server with Ctrl-C.

```

In above screen python web server started and now open browser and enter URL as <http://127.0.0.1:8000/index.html> and press enter key to get below page

Proposed Method



In above screen click on 'New User Register Here' link to get below user signup screen



In above screen user is entering sign up details and give valid EMAIL ID to get OTP password and then press button to complete sign up and get below page



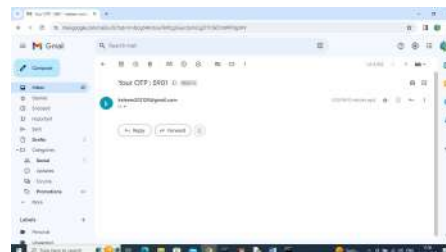
In above screen user signup completed and now click on 'User Login' link to get below page



In above screen user is login and after login will get below OTP page



Above OTP we can receive in given email at sign up time



In above screen 5901 is the OTP which has to enter in OTP validation page like below screen



In above screen after entering OTP then press button to get below page



In above screen click on 'Upload Dataset' link to get below page



In above screen select and upload dataset and then click on “open” and “Submit” button to load and process dataset and then will get below page



In above screen can see dataset loaded and processed and now click on ‘Run Existing’ link to run existing algorithms and then will get below output



In above screen existing SVM and Naïve Bayes training completed and can see SVM got 62% and Naïve Bayes got 68% accuracy and can see other metrics also and now click on ‘Run Auto Encoder’ link to run propose algorithm and then will get below page



In above screen can see existing and propose algorithm performance and now click on ‘Run Extension LSTM’ algorithm link to get below page



In above screen extension LSTM got 100% recall which is higher than existing and propose algorithms and now click on ‘Graph’ link to get below comparison graph



In above graph x-axis represents algorithm names and y-axis represents accuracy and other metrics in different colour bars and in all algorithm extension got high recall.

Conclusion

1. In conclusion, detecting web attacks through intelligent systems like ML and DL offers a viable and effective approach compared to traditional techniques. These methods enable the automation of threat detection, improving response times and reducing false positives.
2. The comparative analysis highlights the strengths and limitations of each algorithm, guiding cybersecurity practitioners in selecting appropriate models based on their specific needs and resource constraints.
3. Deep learning models, while computationally expensive, demonstrate superior performance in complex attack scenarios. However, simpler ML models may still be preferable in resource-limited environments due to their lower training and inference costs.
4. Future work may include integrating these models into real-time detection systems, improving model interpretability, and extending the analysis to other domains such as IoT or mobile platforms, thereby enhancing the scope and applicability of the research.

References

1. Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). **A deep learning approach to network intrusion detection.** *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(1), 41-50.
2. Kim, Y., Kim, W., & Kim, H. K. (2020). **A novel hybrid intrusion detection method integrating anomaly detection with misuse detection.** *Expert Systems with Applications*, 186, 115002.
3. Moustafa, N., & Slay, J. (2015). **UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set).** *Military Communications and Information Systems Conference (MilCIS)*, IEEE.
4. Tavallaei, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). **A detailed analysis of the KDD CUP 99 data set.** *Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications*.
5. Yin, C., Zhu, Y., Fei, J., & He, X. (2017). **A deep learning approach for intrusion detection using recurrent neural networks.** *IEEE Access*, 5, 21954–21961.
6. Vinayakumar, R., Soman, K. P., & Poornachandran, P. (2017). **Applying convolutional neural network for network intrusion detection.** *Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI)*.
7. Dhanabal, L., & Shantharajah, S. P. (2015). **A study on NSL-KDD dataset for intrusion detection system based on classification algorithms.** *International Journal of Advanced Research in Computer and Communication Engineering*, 4(6), 446-452.
8. Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). **A deep learning approach for network intrusion detection system.** *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*.
9. Berman, D. S., Buczak, A. L., Chavis, J. S., & Corbett, C. L. (2019). **A survey of deep learning methods for cyber security.** *Information*, 10(4), 122.
10. Alrawashdeh, K., & Purdy, C. (2016). **Toward an online anomaly intrusion detection system based on deep learning.** *IEEE International Conference on Machine Learning and Applications (ICMLA)*.
11. Ahmad, I., Basher, M., Iqbal, M. J., & Rahim, A. (2018). **Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection.** *IEEE Access*, 6, 33789–33795.
12. Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). **Toward generating a new intrusion detection dataset and intrusion traffic characterization.** *ICISSP*, 1, 108–116. (*CICIDS2017 Dataset*)
13. Sommer, R., & Paxson, V. (2010). **Outside the closed world: On using machine learning for network intrusion detection.** *IEEE Symposium on Security and Privacy (SP)*.
14. Kwon, D., Kim, J., & Kim, J. (2019). **Deep learning-based anomaly detection system for discovering web attacks.** *IEEE Access*, 7, 183527–183536.
15. Goodfellow, I., Bengio, Y., & Courville, A. (2016). **Deep Learning.** MIT Press.