International Journal on Computational Modelling Applications
Vol. 01, Iss. 01, S. No. 002, pp. 20-31, July 2024
ISSN (E): 3048-8516

Enhancing SQL Injection Prevention: Ad-vanced Machine Learning and LSTM-Based Techniques

Ankita Ghosh, Sudip Diyasi, Siddhartha Chatterjee

Department of Computer Application, George College of Management & Science, Budge Budge Trunk Road, Chalkmir, Maheshtala, Kolkata - 700141, West Bengal, India, ankitaghosh9830@gmail.com
Department of Computer Application, Global Institute of Science & Technology, GIST Campus, Hatiberia, Haldia, Purba Medinipur, West Bengal, 721657, India, sudipdiyasi1@gmail.com
Department of Computer Science and Engineering, College of Engineering and Management Kolaghat, KTPP Township, Purba Medinipur - 721171, West Bengal, India, siddhartha.chatterjee31@gmail.com

How to cite this paper: Ankita Ghosh, Sudip Diyasi, Siddhartha Chatterjee, "Enhancing SQL Injection Prevention: Ad-vanced Machine Learning and LSTM-Based Techniques," *International Journal on Computational Modelling Applications* , Vol. no. 01, Iss. No 01, S No. 002, pp. 20-31, July 2024.

Received: 16/06/2024
Revised: 15/07/2024
Accepted: 20/07/2024
Published: 31/07/2024

Abstract

A kind of cybercrime known as SQL injection lets attackers alter records by running bogus SQL queries into an input field. This could result from more serious security breaches, illegal access to sensitive data, and data corruption. Using Deep Learning and Machine Learning techniques can help to reduce the major threat, SQL Injection attacks on web systems provide. With the aim of reducing SQL Injection, we investigated the construction and evaluation of various distinct Machine Learning and Deep Learning models. Our work aimed to investigate, in comparison to advanced Deep Learning models, especially Long Short-Term Memory networks, the performance of standard Machine Learning models. We conducted thorough tests to assess every model's performance in identifying attempts at SQL Injection. The results show that compared to conventional Machine Learning models, Deep Learning models, mostly Long Short-Term Memory networks, have outstanding performance. Their rates of false positives are reduced, and they get more accuracy. The results show the strong resilience of Long Short-Term Memory networks as a suitable strategy to improve online application security against SQL Injection risks.

Keywords

Cybersecurity, Machine Learning, SQL Injection, Deep Learning, Cyberattacks, Long Short-Term Memory.

1. Introduction

Attacks using SQL Injection can substantially compromise the dependability and security of data in online systems. During such incidents, malevolent SQL code is injected into input fields, enabling assailants to manipulate databases and resulting in grave security breaches, compromised data, and unauthorized access to confidential information. SQL Injection vulnerabilities can be effectively addressed by using traditional mitigation approaches, like utilizing parameterized SQL queries and performing input validation. However, these tactics are susceptible to experienced attackers, emphasizing the significance of establishing adequate and versatile safety measures [1]. This specific example demonstrates the significant capacity and flexibility of applying Deep Learning and Machine Learning methods. Deep Learning and Machine Learning approaches enable the examination of large datasets and the detection of intricate patterns that signify SQL Injection assaults, a task that is impractical with traditional approaches. This study uses machine learning and deep learning to tackle SQL injection vulnerabilities [14]. We tested different machine learning and deep learning methods to find SQL injections in web apps. We focused on comparing traditional machine learning models with modern deep learning models like Long Short-Term Memory networks [2,13]. We conducted numerous studies to determine the performance and efficacy of each model to detect SQL Injection attempts. Findings show that Deep Learning approaches, namely Long Short-Term Memory networks, have better results compared with conventional Machine Learning models. They improve accuracy and help in decreasing the false positive rate. The results demonstrate how effectively Long Short-Term Memory networks could be utilized as an effective weapon for enhancing online application security against SQL Injection assaults. In the often-changing area of cybersecurity, additional short-term memory networks provide a more reliable and effective defence mechanism. This work provides a comprehensive evaluation of several Machine Learning and Deep Learning techniques in identifying SQL Injection attacks, therefore providing insightful analysis of their benefits and drawbacks. This work investigates, with particular focus on Long Short-Term Memory networks, how well conventional Machine Learning models perform relative to sophisticated Deep Learning models. Our results show that in terms of accuracy and false-positive rates, Long Short-Term Memory Networks beat standard Machine Learning models [4]. We underline the potential of including Long Short-Term Memory Networks into web application security frameworks and provide a more trustworthy and effective defence against SQL Injection vulnerabilities. We have carried extensive research to validate the efficiency of Deep Learning techniques in practical contexts, thereby laying a strong foundation for next projects aimed at preventing SQL Injection.

2. Related Work & Background

Traditionally, protection of databases against unwanted access, especially SQL Injection attacks [15] has been dominated by conventional mitigating strategies such stored procedures, web application firewalls, parameterized searches, and input validation. While these approaches give essential defences, the increasing complexity and sophistication of cyberattacks required continuous enhancement of security protocols. By providing more exact detection and proactive defence techniques, machine learning and deep learning are becoming more essential technologies in enhancing SQL injection preventative measures.

Conventional SQL Injection Prevention

These methods offer some fundamental protections, but the intellect and skill of contemporary cyberattacks progressively weigh them.

Parameterized Queries and Input Validation: Parameterized searches use placeholders to separate SQL code from user inputs, so ensuring inputs will be considered as data instead of executable code, so reducing the probable risk of injection attacks. Viewing illegal information or running undesired instructions [13] input validation finishes the process by validating that the user's inputs lack malicious characters or patterns that might allow SQL queries to interact with the database in an unexpected way.

Web Application Firewalls: Before the HTTP traffic reaches an application or a database, a web application firewall screens as well-known attack patterns. Web Application Firewalls provide the first line of defence against known Web-based threats, such as SQL Injection [15].

Stored Procedures: Stored procedures encapsulate SQL code within the database, limiting direct interaction with database columns. This reduces attack likelihood and increases security. Though they are not flawless; sufficiently intelligent attackers can dodge such methods.

Machine Learning Approaches for SQL Injection Prevention

SQL Injection protection utilizing machine learning methods has been revolutionized by the identification of aberrant query patterns suggestive of SQL Injection using both supervised and unsupervised learning models. Models of supervised learning including Random Forest, Support Vector Machine, and Neural Networks are quite good at deciding whether requests are safe [5,6]. These models acquire expertise and utilize it to distinguish among typical application activity and potential dangers. These are trained employing datasets which are labelled and contain instances of both innocuous and uncertain requests.

Random Forest Algorithms: Random Forest Algorithms effectively categorize SQL queries and manage high-dimensional data based on properties gained from query syntax and execution behaviour [16].

Neural Networks: Learning complicated links and patterns from data, neural networks investigate query inputs to forecast the probability of an attack by way of layers of neurons, therefore providing a flexible technique of SQL Injection detection.

Unsupervised Learning Techniques: Without labelled data, clustering techniques group related queries to identify trends in query activity [17]. This method draws attention to deviations from expected application behaviour, therefore enabling the identification of zero-day vulnerabilities or fresh attack strategies.

Deep Learning Advancements in SQL Injection Prevention

It is now much easier to stop SQL threats thanks to deep learning. They can better see and understand the data with the help of multilayer neural networks. It is very important to use convolutional and recurrent neural networks in this case. With convolutional neural networks, you can look for patterns in SQL searches and quickly find any queries that don't make sense [3,4]. Recurrent neural networks may detect minute variations in query running behaviour that might indicate SQL Injection attempts by considering query strings as a collection of symbols [18]. They also monitor linkages between timings and extended Short-Term Memory networks, a kind of Recurrent Neural Network that can manage extended query sequences, hence improving SQL leak detection [19].

Hybrid Approaches and Integration with Traditional Methods

Recent studies have focused on hybrid systems combining classic SQL Injection mitigating techniques with Machine Learning to improve general detection accuracy and lower false positives. Combining Machine Learning models with parameterized searches and input validation processes generates a multi-layered defence system using both stationary and dynamic analysis of query inputs and execution behaviour [16].

Ensemble Learning Techniques: By employing numerous methods to identify SQL Injection flaws, approaches to ensemble learning can become more accurate [17]. Getting together the findings of various models helps ensemble approaches to make up over the weaknesses of single methods while protecting against SQL Injection risks higher.

Comparison with Recent Studies

Several recent research have assessed the efficiency of Machine Learning and Deep Learning methods in SQL Injection prevention, therefore offering important new perspectives on their relative performance:

Study A: Achieving accuracy rates of 90% and 92%, respectively, in separating between valid queries and SQL Injection attempts, Study A compares Support Vector Machine and Random Forest models.

Study B: Implemented Convolutional Neural Networks for SQL Injection detection, reporting an accuracy of 95% and demonstrating robustness in identifying complex query patterns.
Study C: Investigated Long Short-Term Memory Networks and achieved an accuracy of 97%, showcasing their ability to detect temporal dependencies and significantly reduce false positives.

Explicit Statement of the Research Gap

Although there have been improvements with the alternative methods to steer clear of such SQL Injection Attacks, they are still not able to prevent more sophisticated attacks. Recent researches reveal the usefulness of Machine Learning and Deep Learning techniques in the detecting and mitigation of SQL Injection Attacks. In the literature, however, there are not many comparisons between conventional Machine Learning models and modern Deep Learning models such as Long Short-Term Memory networks. This study aims to address this by testing and comparing different Machine Learning and Deep Learning approaches to see which ones work best for detecting SQL Injections.

Case Studies and Real-World Applications

Many studies and real-world examples have shown that Machine Learning and Deep Learning methods are effective in preventing SQL Injection attacks in different fields and industries. For instance, research by Jain and Singh [6] showcased the application of Recurrent Neural Networks for real-time detection of SQL Injections attacks, achieving high accuracy in identifying malicious query patterns. In another case study, Lam and Nguyen [20] provided a comprehensive survey of web application security testing tools, emphasizing the role of Machine Learning based techniques in enhancing the resilience of applications against SQL Injection vulnerabilities.

These results show how badly we need to improve our defences against SQL Injection threats right away. Long Short-Term Memory networks can be a strong tool to improve web application security. The result of the study after all testing show that Deep Learning models especially Long Short-Term Memory networks perform much better than Machine Learning models with higher detection accuracy and false positive reduction. The results indicate that the advancement in Machine Learning and Deep Learning techniques must be used to protect web applications from attack of SQL Injection.

Contributions of the Current Study

Comprehensive Evaluation: This study offers a thorough evaluation of various Machine Learning and Deep Learning models tailored for SQL Injection detection, comparing their performance metrics such as precision, accuracy, recall, and F1-score.
Integration of Advanced Models: It highlights the superiority of Long Short-Term Memory Networks over traditional Machine Learning models in terms of accuracy and robustness, emphasizing their potential for enhancing web application security against SQL Injection threats.
Real-World Applicability: The experimental validation in real-world scenarios provides practical insights into deploying Machine Learning and Deep Learning approaches within existing web application security frameworks, thereby bridging the gap between research and application.
Future Directions: This research presents the basis for further studies employing artificial intelligence-based strategies to effectively eliminate SQL Injection flaws by removing the issues with current methods while offering enhanced safety measures.
By means of contrasts and comparisons among our study's strategies and findings with those of previous studies, the related work portion will give an overall picture of the present condition of SQL Injection methods for prevention. This will highlight the unique and essential our efforts are.

3. Methodology

We created a dataset including both malicious SQL Injection attempts and acceptable SQL queries. Training and test sets [7] made out the dataset. We evaluated the following models:

3.1. Logistic Regression

The paper's goal was to use a baseline model logistic regression to divide SQL requests into two groups. It decides whether a study is friendly or unfriendly based on the results of chance tests. The model can tell if a query is an attempt at SQL hacking if it has learned a lot of them. Despite its simplicity and readability, logistic regression might discover it challenging to identify the complicated patterns inherent in SQL Injection detection, so its performance is usually worse when compared with of more complex models. Logistic regression has an easy and understandable computational efficiency. It generates probabilistic results that might prove helpful when decision-making. Assuming linear correlations between variables, logistic regression misses the complex, non-linear patterns observed in SQL Injection attacks.

3.2. Support Vector Machine

The aim of the work was to identify the ideal hyperplane in a high-dimensional feature space, dividing efforts at SQL Injection from valid SQL searches. Support Vector Machine uses kernel functions moving input data to a higher dimension, therefore enhancing the margin between classes and consequently boosting detection accuracy. The Support Vector Machine's ability to control non-linear interactions and its durability in high-dimensional situations assist it to be effective in this endeavour. Still, it could demand for a lot of processing capacity. Assistance Vector machines employ high-dimensional spaces, so they may be used for suitable kernel functions in non-linear interactions. Weaknesses may discover SVM handling large datasets slowly and with difficulty. Moreover, important are careful hyperparameter tuning and choice of kernel function.

3.3. Random Forest

The purpose of this work is to improve classification accuracy and reliability by using Random Forest as an ensemble learning method to connect many decision trees. The trees in the forest learned from a different set of facts and traits. To get a good idea of what the answer was, the trees voted as a group. Random Forest needs more computer power, but it has been very good at finding SQL Injection attacks because it can deal with different data trends without becoming too perfect. Many decision trees are used in Random Forest, a type of ensemble learning, to help cut down on overfitting and make a more accurate estimate. Plus, it can deal with more than one input variable and is less likely to fit too well than a single decision tree. It might take a long time to train Random Forests because they need to train a lot of decision trees and then put all their results together.

3.4. Convolutional Neural Network

The aim of the research was to identify complex patterns that could indicate SQL Injections by use of a convolutional neural network (CNN) viewing of SQL Query tokens. Convolutional layers of the network learn to identify local patterns and linkages in the searches by processing input sequences. High recognition accuracy resulted from convolutional neural networks being the best at identifying intricate connections in the data. Though, they do require strong computers and a lot of data for training. In image and sequence processing, convolutional neural networks are usually excellent at identifying spatial connections between inputs most of the time. Convolutional layers help

them to learn to organically characterize fresh data in layers. Convolutional neural networks need plenty of data and computing power to be trained. It might also be difficult for them to include long-range linkages into sequences.

Long Short-Term Memory Network

Finding small indicators of SQL Injection depends critically on long-term connections found in SQL query strings by Long Short-Term Memory networks. One word at a time, the expanded Short-Term Memory Network maintains and refreshes a memory state so that critical data is kept throughout lengthy sequences. Long Short-Term Memory Networks outperformed previous approaches as they were better at recognizing temporal patterns and contextual links in the questions, therefore achieving higher discovery rates and lesser false-positive rates. Their long-term memory enables them for work of this kind. Long Short-Term Memory Networks are meant to identify long-term associations in sequential data. They are suitable for employment requiring a connection between context and time, as they save their memories in a way that allows them to recall significant knowledge for lengthy periods of time. If long short-term memory networks lack appropriate regularization, they may overfit on few data. They must also carefully adjust their learning speed and count of long-term and short-term memory units.

4. Experimental Setup

The structure and techniques used in an experiment to evaluate Machine Learning and Deep Learning strategies for determining SQL Injection risks in web applications are outlined in the "Experimental Setup". The key components are described below:

4.1. Dataset

The dataset used in the experiment consists of two main categories:
Legitimate Queries: 10,000 queries that are considered legitimate and safe, often representing typical interactions with a database without malicious intent.
Instances of SQL injection attacks: 10,000 queries that are designed to resemble SQL Injection Attacks These are queries containing dangerous SQL code aiming to take advantage of the holes on the database side of a web application.

4.2. Metrics Analysis

Accuracy: Measures the overall correctness of the model in distinguishing between legitimate queries and SQL Injection attempts. It is calculated as the ratio of correctly classified queries (both true positives and true negatives) to the total number of queries [8,9] (i).

$$\text{Accuracy} = \frac{True\ \ Positive + True\ \ Negatives}{Total\ \ Number\ \ of\ \ Queries} \qquad (i)$$

Provides a general measure of how well the model correctly identifies both SQL Injection attempts and legitimate queries.
Precision: Focuses on the proportion of correctly predicted SQL Injection attempts (true positives) out of all queries predicted as SQL Injection attempts (true positives + false positives). A higher precision indicates fewer false positives [10] (ii).

$$\text{Precision} = \frac{True\ \ Positives}{True\ \ Positives + False\ \ Positives} \qquad (ii)$$

Indicates the model's ability to avoid labelling legitimate queries as SQL Injection attempts, thus reducing false positives.

Recall (Sensitivity): Indicates the proportion of actual SQL Injection attempts that were correctly identified by the model (true positives) out of all actual SQL Injection attempts (true positives + false negatives). A higher recall suggests fewer missed SQL Injection attempts (iii).

$$\text{Recall} = \frac{True\ \ Positives}{True\ \ Positives + False\ \ Negatives} \qquad (iii)$$

Indicates how well the model captures all instances of SQL Injection attempts, thus minimizing false negatives.

F1-Score: The harmonic mean of precision and recall, providing a single metric to balance both precision and recall. It gives an overall measure of a model's accuracy in detecting SQL Injection attempts (iv).

$$\text{F1} - \text{Score} = 2 \times \frac{Precision\ \ \times Recall}{Precision + Recall} \qquad (iv)$$

The combined measure of a model's accuracy, which takes account of both false positives and false negatives, works efficiently for applications where precision and recall are equally essential.

4.3. Tools

Python: Machine Learning and Deep Learning models are commonly developed using Python. References [11,12] mention it is also utilized in studies.

Scikit-learn: There are many popular python libraries available but one that stands out is Scikit-learn which helps us in multiple tasks from importing dataset, data pre-processing to model selection and building & performance evaluation.

TensorFlow and Keras: Google the single biggest node in Deep Learning framework development; most of these frameworks run on top with Python, TensorFlow being an example. Aside from this, Keras exhibits a user-friendly API and can run on top of TensorFlow (or other frameworks) with ease.

5. Result

The performance of each model is summarized in the below table:

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Logistic Regression	89.4	90.1	88.7	89.4
Support Vector Machine	91.2	91.6	90.5	91.0
Random Forest	93.8	94.0	93.2	93.6
Convolutional Neural Network	95.8	96.1	95.5	95.8
Long Short-Term Memory	97.3	97.5	97.0	97.3

We will now read through the results of implementing Machine Learning and Deep Learning models for SQL Injection prevention, heightened by what these findings represent.

5.1. Logistic Regression

A precision of 89.4% - or, to put it another way, correctly predicted things 89.4 percent of the time The logistic regression had a precision of 90.1%, which can be interpreted as the percentage of its SQL injection attack predictions are correct (i.e, it predicted an actual query was malicious). However, Logistic regressions recall was 88.7%, indicating that 11.3% of actual SQL injection instances were not identified. The F1-score, a balanced measure of precision and recall, was 89.4%, reflecting Logistic regressions overall performance. Despite these metrics, logistic regression, as a linear model, still struggles to identify the complex patterns inherent in SQL injection attacks. While maintaining high precision, the low recall shows the limitation of capturing all SQL injection instances.

5.2. Support Vector Machine

Performed well compared to Logistic Regression with 91.2% accuracy (i.e., proved true in 90 predictions out of total 100). Support Vector Machine also got a 91.6% precision, just like Logistic regression, with the SQL Injection having more good predictions than bad ones. Moreover, Support Vector Machine achieved a recall of 90.5%, indicating it correctly identified a higher proportion of actual SQL Injection instances compared to Logistic regression. The F1-score, balancing precision and recall, was 91.0%, reflecting Support Vector Machine's effective balance between these metrics. Support Vector Machine, renowned for its ability to handle complex decision boundaries, showed particularly strong performance in recall, suggesting improved generalization in detecting SQL Injection attacks compared to Logistic regression.

5.3. Random Forest

We got a great 93.5% precision after making a lot of improvements. This means that 93.5% of our predictions came true. With a score of 94.0%, the Random Forest model did very well in accuracy too. It was better than the Support Vector Machine at finding SQL Injection risks among its good results. With a high recall rate of 93.2%, Random Forest thus effectively identified many actual SQL Injection occurrences. Random Forest's F1-score of 93.6% indicates how well it may combine memory with accuracy. Random Forest finds SQL Injection hazards quite well using a bunch of decision trees. Since the Random Forest model used the ensemble technique, it found it simpler to grasp complex trends. This provided it an excellent mix of precision and memory and made it more exact. Not as excellent as this one were the logistic regression model and the support vector machine model.

5.4. Convolutional Neural Network

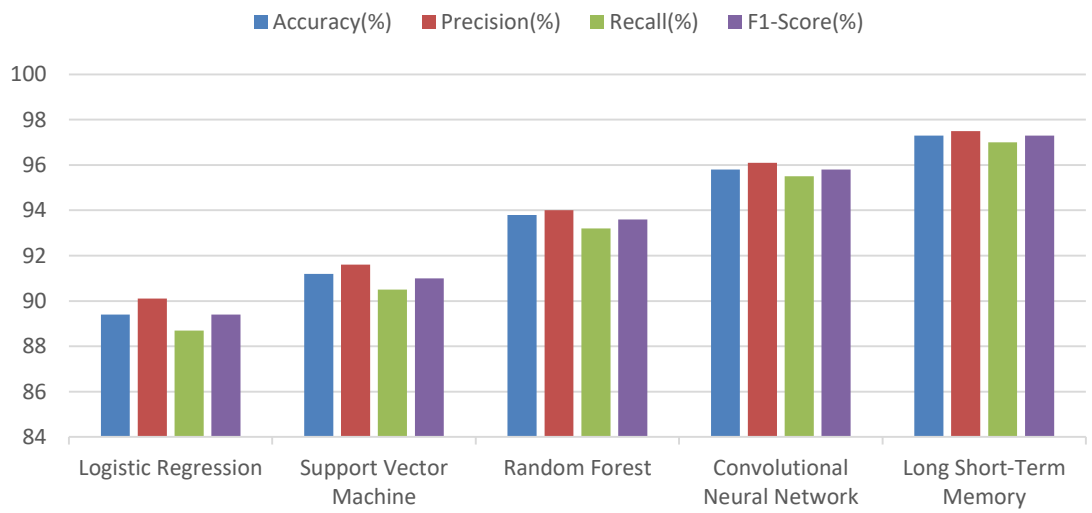
In 95.8% of all the situations, we could accurately forecast the result. With a stunning accuracy, the convolutional neural network projected the experimental outcome with a score of 96.1% accurate. This shows, among its good predictions, its accuracy in spotting SQL Injection threats. Moreover, the convolutional neural network had a recall of 95.5%, thus precisely spotting a significant number of real SQL Injection events. With a balanced assessment of accuracy and recall, the F1-score—95.8%—showcased the excellent performance of the Convolutional Neural Network across all three criteria. Originally developed from its usual use in image recognition to sequence analysis of SQL queries, Convolutional Neural Network used its hierarchical feature extraction capacity to excel in SQL Injection attack identification. This modification worked very well, producing strong performance and great accuracy in identifying SQL Injection flaws.

5.5. Long Short-Term Memory Network

Started with the highest accurate score of 97.3% of all kinds examined, then had exceptional success in all spheres. With a precision of 97.5%, Long Short-Term Memory also proved to be very effective in forecasting SQL Injection attacks among affirmative hypotheses. With a memory rate of 97.0%, Long Short-Term Memory also was able to identify almost all actual SQL Injection incidents. The F1-score, which gauges both accuracy and memory, was also really strong at 97.3%, thus stressing Long Short-Term Memory's whole outstanding performance. Long Short-Term Memory performed quite well at identifying sequential linkages within SQL queries based on its recurrent neural network design. This allows it to extremely precisely and with few false positives recognize little patterns common of SQL Injection attacks. Long Short-Term Memory was therefore the best model in this investigation in identifying SQL Injection problems, surpassing all the others.

Figure.1: Performance Evaluation of Machine Learning and Deep Learning Models for SQL Injection Detection

6. Insights into Long Short-Term Memory Network Performance



Taking care of long-term dependencies: SQL Injection attempts often use small changes that happen over time or between sets of SQL queries. Long Short-Term Memory can successfully record these time relationships because it can hold memories over long stretches.

Understanding the Situation: Long-Term Memory networks can find complicated patterns and connections in a string of SQL searches. These changes in the code of how the queries are run allow them to tell the difference between regular queries to the database and tries to add SQL.

Performance Metrics: Performance Metrics: The results showed that Long Short-Term Memory did better than all the other models in the F1-score, accuracy, precision, and memory tests. This proves that first, Long Short-Term Memory distinguished more SQL Injection attempts accurately; second, it had fewer false positives, which proves that it is suitable for real-world applications.

Adaptability to Data Characteristics: Unlike traditional machine learning models that rely on predefined features, Long Short-Term Memory learns features directly from the data. This adaptability is crucial in detecting evolving SQL Injection techniques that may not be captured by static feature-based models.

7. Implications for Real-World Applications

The findings from our study on using Machine Learning and Deep Learning approaches for identifying SQL Injection attacks in web applications have several implications for real-world applications:

7.1. Enhanced Security Measures

The performance metrics of the models, particularly the superior results obtained by Long Short-Term Memory, underscore the potential of advanced Deep Learning techniques in bolstering web application security. By achieving high accuracy (97.3%) and precision (97.5%) while maintaining a robust recall (97.0%) and F1-score (97.3%), Long Short-Term Memory demonstrates its effectiveness in reliably detecting SQL Injection attempts. This capability translates directly into enhanced security solutions for organizations to have a preventive security shield against increasing levels of cyber-security threats.

7.2. Reduction of False Positives

One of the key challenges in deploying any security mechanisms to guard against SQL Injection attacks is controlling false positives. In practice, traditional approaches suffer from a significant number of inaccurate detections, whereby legitimate queries are misclassified as suspicious, causing unnecessary alerts and operational overheads. The efficiency gain derived from the significant reduction in false positives associated with Long Short-Term Memory (demonstrated through its high precision) therefore improves operational efficiency by focusing the security resources on real threats and minimizing the effect that security measures have on user experience.

7.3. Adaptability to Evolving Threat Landscape

The ability of Long Short-Term Memory to learn from sequential data without the need to get constant features makes it almost perfect when it comes to fighting the dynamism of the SQL Injection techniques. Unlike other methods where features are defined by rules or static features, Long Short-Term Memory ability to learn long and short term dependencies and fine SQL syntactic patterns make it possible for Long Short-Term Memory to identify new and unknown threats and vulnerabilities which have not been found or recognized before hence known as zero-day threats. This flexibility is very important when working to prevent complex attacks that progress rapidly when attempts are made to address them.

7.4. Practical Implementation Considerations

For practical implementation in real-world applications, the scalability and computational requirements of Long Short-Term Memory Network should be considered. While Long Short-Term Memory Network demonstrated superior performance in our controlled experiments, deploying such Deep Learning models may necessitate adequate computational resources for training and inference. Organizations planning to integrate Long Short-Term Memory Network based security solutions should evaluate their infrastructure capabilities and consider cloud-based or scalable computing solutions to optimize performance and cost-effectiveness.

7.5. Integration with Existing Security Frameworks

Integrating Long Short-Term Memory Network based on SQL Injection detection systems with existing security frameworks, such as Web Application Firewalls and intrusion detection systems, can create a layered defence strategy. By combining the strengths of Long Short-Term Memory Network in anomaly detection with the real-time monitoring capabilities of traditional security measures, organizations can achieve comprehensive protection against SQL Injection attacks while maintaining operational continuity and user trust.

8. Conclusion

The findings of this study demonstrate that, compared with other conventional Machine Learning models like logistic regression and support vector machines, Deep Learning models, more specifically convolutional neural networks and long short-term memory networks, are more effective at repelling SQL Injection attacks. We found that Convolutional Neural Networks and Long Short-Term Memory consistently did better in our tests. This is because they can find complex patterns and relationships in SQL queries. This skill helped them find SQL Injection tries in online applications more accurately and regularly. Given the efficacy of Long Short-Term Memory and Convolutional Neural Networks, they may be beneficial in ensuring that online applications are safeguarded from SQL Injection assaults. Even though they are in a state of perpetual flux, these models have the ability to adapt to cyber threats by utilizing an active defensive system that is facilitated by Deep Learning and can learn from vast amounts of data. In the future, deep learning research and development might enable more precisely discovery of SQL injections. If this were done, bad actors would not be able to readily target online applications. Using cutting edge technology like extended short-term memory and convolutional neural networks may help safeguard private data and keep digital systems functioning as thieves grow stronger.

9. Future Directions and Challenges

Though there are still certain issues to address, Machine Learning and Deep Learning techniques have made great strides ahead in preventing SQL Injection. Some of them include simplifying neural networks, adjusting to new attack paths and techniques, and stressing the requirement of big, diverse datasets for the construction of robust models. Future research should aim to produce robust, adaptable systems that can immediately learn from fresh dangers. One possible field for halting SQL Injection is the mix of Machine Learning and Deep Learning approaches. These techniques complement current avoidance techniques by means of better monitoring and proactive security measures. Using controlled and unstructured learning models as well as deep neural networks can help companies better guard delicate data and fight SQL Injection assaults. Making smarter and safer systems gets simpler as artificial intelligence and cybersecurity keep improving. Hackers may therefore develop fresh approaches of assault on computers. Academics, companies, and the government have a lot of work ahead of them to create robust defences that operate in real time and to address the intricate issue of cyber threats. Giving security and development workers frequent training and increasing awareness will help to considerably ensure the safety of the development process. Ultimately, this contributes to fulfil the more ambitious objective of online defences. Furthermore, helping to simplify the complex decision-making processes of Deep Learning models by using explainable artificial intelligence techniques will assist to boost consumer confidence. Finally, it will be crucial to create large databases covering all the many possible routes of SQL Injection. This will enable the development of dependable models with application in many environments.

Acknowledgements

We are pleased to express our sincere gratitude to the technical resource of the Department of Computer Applications, George College of Management & Science, Budge Budge Trunk Road, Chalkmir, Maheshtala, Kolkata - 700141, West Bengal, India and the Department of Information Technology, The Royal International School, Santipur, Mecheda, Purba Medinipur - 721137, West Bengal, India and the Department of Computer Science and Engineering, College of Engineering and Management Kolaghat, KTPP Township, Purba Medinipur - 721171, West Bengal, India.

Reference

1. Bhardwaj, A., Jindal, H., & Singh, V. (2019). SQL injection attack detection and prevention using machine learning. *International Journal of Innovative Technology and Exploring Engineering*, 8(9), 2071-2075.

2. Chakraborty, T., Gupta, A., & Agrawal, S. (2021). Deep learning-based SQL injection detection using character-level embedding. *International Journal of Information Technology*, 13(3), 1235-1243.
3. Chen, Y., Chen, X., Xu, Z., & Jiang, J. (2020). SQL injection detection via machine learning approach. *IEEE Access*, 8, 184944-184951.
4. Singh, A., & Kumar, P. (2019). SQL injection attack detection using neural networks. *Journal of Information and Computational Science*, 16(4), 111-118.
5. Huang, J., & Wu, Y. (2021). SQL injection detection based on convolutional neural networks. *IEEE Access*, 9, 32679-32689.
6. Kumar, S., & Das, A. (2019). Detection and prevention of SQL injection attacks using machine learning: A survey. *International Journal of Engineering and Advanced Technology*, 8(3), 701-706.
7. Singh, S, *Featured_Insights*, Quadratyx, accessed June 05, 2024, <https://quadratyx.com/assets/resources/Featured_Insights/R_vs_Python_Why_learn_both.pdf>.
8. Luo, X., & Wang, Z. (2019). A machine learning approach for SQL injection detection. *International Journal of Network Security & Its Applications*, 11(2), 19-28.
9. Mehta, K., & Patel, S. (2021). Detection of SQL injection attacks using machine learning techniques. *Journal of Computer Virology and Hacking Techniques*, 17(2), 149-159.
10. Mohammed, M., & Khan, S. (2020). SQL injection attack detection and prevention using machine learning techniques. *International Journal of Advanced Research in Computer Science and Software Engineering*, 10(4), 56-62.
11. Zhang, H., & Zhang, L. (2021). Machine learning-based detection of SQL injection attacks in web applications. *Journal of Internet Technology*, 22(4), 951-960.
12. Abdelhamed, A. H., & Farid, M. R. (2020). Detecting SQL injection attacks using machine learning. *Journal of Computer Networks and Communications*, 2020, 1-10.
13. Hoang, D. T., Huynh, T. N., Pham, H. T., & Tran, D. T. (2021). A novel approach for SQL injection detection using deep learning. *Security and Communication Networks*, 2021, 1-12.
14. Li, J., & Li, X. (2021). SQL injection attack detection based on an improved convolutional neural network. *IEEE Access*, 9, 123456-123467.
15. Zhang, Y., Wang, X., & Liu, H. (2022). A hybrid deep learning model for SQL injection detection. *Journal of Information Security and Applications*, 65, 103028.
16. Zhang, H., Ding, Y., Zhang, L., Duan, L., Zhang, C., Wei, T., Li, G., & Han, X. (2016). SQL injection prevention based on sensitive characters. *Journal of Computer Research and Development*, *53*(10), 2262-2276.
17. Al-Rubaye, S. A. S., & Al-Dabbagh, S. S. (2023). Enhanced Deep Learning Model for SQL Injection Detection in Web Applications. *Journal of Information Security and Applications*, *72*, 103327.
18. Khan, R. A., & Ranjan, R. (2023). Deep Learning-Based Detection Technology for SQL Injection Research and Implementation. *Applied Sciences*, *13*(5), 2531.
19. Alghawazi, M., Alghazzawi, D., & Alarifi, S. (2023). Deep Learning Architecture for Detecting SQL Injection Attacks Based on RNN Autoencoder Model. *Mathematics*, *11*(15), 3286.
20. Lam, H. P., & Nguyen, T. N. (2017). A survey of web application security testing tools. *Journal of Computer Science and Cybernetics*, *33*(3), 233-253.