International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Effective Detection and Mitigation of Browser Hijacking Using Deep Learning Models

Sambhav Patil[1], Chandra Sekhar Dash[2]

[1]School of Computer Science and Engineering, Bundelkhand University, Jhansi
[2]Senior Director, Governance, Risk and Compliance Ushur Inc, Dublin, CA, USA

Abstract

Deep learning algorithms have been employed in the detection and prevention of browser hijacking and this research paper seeks to compare the efficiency of the different deep learning algorithms. Browser hijacking is a serious form of cybercrime that affects major internet browsers, and entails the changing of a browser's settings or behavior without the consent of a user, and generally results in detrimental effects on the privacy and security of a user. To address this challenge, we evaluate five deep learning models: Some of the major types of neural networks are Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), Autoencoders and Graph Neural Networks (GNNs). In each model, the training and testing data set included the network traffic records, browser activity recordings, screen shots and the system logs. Specifically, for the evaluation of these models the following parameters of detection: accuracy, false positive rate, false negative rate, precision, recall, the F1 factor, the AUC score. I found CNN model to have the highest detection accuracy of 93. 5% and the highest AUC score of 0. 96 showing that CNN was more effective in detecting the hijacking and the benign activities. Other represented algorithms, LSTM and GNN also showed a high level of accuracy with values 91. 2% and 90. 3% accordingly. The accuracy of the RNN model was satisfactory yet its false positive as well as false negative numbers were lower. The used Autoencoder model was most suitable for data anomaly detection but it presented the lowest accuracy and the highest error. The results point out the merits and demerits of each algorithm and indicate that integration of models can boost total identification capacities. All in all, it is possible to underline that this research provides a significant contribution into investigating the applicability of deep learning methodologies for mitigating browser hijacking and enhances the existing knowledge and concepts in the field of cybersecurity.

Keywords: browser hijacking, deep learning, Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), Autoencoders

1. Introduction

Browser hijacking is a severe form of cyber threat through which a web browser's settings are altered by a third party so as to redirect the users to a number of unwanted web sites, or compromise the users' privacy and their browsing habits as well. This has become more alarming of late because attackers use various ways of bypassing common security measures. The severity of these attacks has grown more complicated, thereby sparking awareness on the development of new detecting and pulling down mechanisms, and with deep learning models being perceived as ideal for analyzing complicated patterns

in data.

Browser hijacking is the act of altering a users' web browser settings without permission for example by changing their homepage, search engine or new tab configuration. This is usually done by either downloading malicious software's or having hidden installations. Overall, browser hijacking can cause mild disorder to severe threat, meaning that a user can experience inconveniences to facing identity theft. For example, users are likely to be redirected to fake sites that can steal personal information or other things are likely to be done based on the user's browsing habits. Such outcomes indicate the severity of the problem and the importance of developing proper mechanisms for its early identification and subsequent prevention.

Previous approaches to detecting hijacked browser mainly employ heuristic signature-based methods, which are based on specific predefined rules or a database of known patterns of specific malicious activities. While these methods have been somewhat effective the following is the disadvantages that they present. Bec enthusiastic vandals employ multiple strategies in their attacks while staying updated with their tactics making it a game of cat and mouse where conventional methods prove incompatible often. In addition, these techniques might have high false alarm rate in that it may indicate that legal behaviours are actually malicious or it may fail to identify new types of threats. The wave-based and heuristic approaches are not very flexible because often they do not account for the changes in the threats for a long time and may not be efficient against new threats of the constantly evolving threat.

While browser hijacking could be dealt with using conventional machine learning techniques, deep learning models, a subfield of machine learning, have a more flexible and effective solution in identifying and preventing this sort of problem as explained below. It should therefore be pointed out that the nature of deep learning as a machine learning method serves it well here since it is capable of inherent multi-dimensional pattern matching. These include Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) which can extract the features on their own from the raw data thus avoiding the need for feature extraction. This auto-extraction of features enables the deep learning model to detect the patterns, which are very hard to detect normally, for hijacking activities. Furthermore, deep learning models are quite flexible, which implies that they can be trained to detect new types of threats in the course of their development, which is a feature that traditional approaches seldom have.

For deep learning to be effectively implemented to detect browser hijacking a clear procedure must be followed. It has several steps, which include data collection, as a starting point. A large and varied corpus is necessary since it should contain all forms of browsers' activity starting from normal ones up to cyberthreats. Some of the data that could be collected may include; network traffic logs, history on different browsers, and how the user interacts with the system. After data have been collected, some preparation is needed in order to analyze the data properly; this step is called data preprocessing. This step involves pre-processing the data where features could be normalized and cleaned before data handling, treatment of missing data and transformation of raw data for deep learning models. The next step comes after the data is prepared and that involves choosing the right deep learning architecture for the task. C CNN) for the spatial and RNN for the sequential type of data, where the choice of model depends on the relative peculiarities of the given dataset and the nature of the hijacking threats under considerations. The training process comes in whereby the model needs to be fed with large sets of data in order to learn when certain behaviors are normal and when they can be considered as malicious. In conclusion, based the current trends of browser hijacking threats, better means of detection and

eradication must be used. More and more, traditional heuristic and signature-based methods can not suffice for a number of reasons, primarily for their inability to scale and adapt to new threats. Consequently, deep learning models become a viable solution by their ability to analyze the patterns or structure of the data and learn about new attack patterns. Thus, by using a clear step-by-step method which is focused on data gathering, its preprocessing, and creating a deep learning model, it can be concluded that deep learning can improve the efficiency of browser hijacking detection and can give a stronger protection against this emerging cybersecurity issue.

2. Literature Review

Currently, improved techniques of detecting browsers hijackers are a clear indication that threats in the cybersecurity environment are continuously evolving as well as growing in intensity, which requires enhanced ways and means of detecting such threats. Although, conventional methodologies are findings it difficult to cope up with new and sophisticated techniques, latest works have shown the potential of machine learning and deep learning methods. This paper aims to present an overview of the selected literature from the year 2022, 2023 and 2024 only to focus on the methodologies created to counter browser hijacking.

A significant work of Zhang et al. is published in 2022 where research was conducted on the integration of ensemble learning methods for browser hijacking detection. It indicates that heuristic and signature-based methods fail to boast high accuracy and, hence, overstate either the number of threats to be found (false positives) or ignore potentially malicious files (false negatives). Mult-classifier integration was suggested by Zhang et al. , where several classifiers such as Decision Tree, Support Vector Machines and Random Forests were integrated in a single set up. Thus, the three individual classifiers were combined in an ensemble model that made enhanced general detection accurate and less sensitive to error. The study also proved the proposed ensemble method to have the ability to acquire a more diverse set of hijacking actions and dynamics while incorporating new attacks. This study clearly showed that it is possible to overcome the disadvantage of single-classifier methodologies through the use of ensemble learning and provided the basis for future work on the use of combined architectures.

The year 2023 is marked by developmental transitions specifically in the areas of deep learning methodology which have investigated different methodologies to detect browser hijacking. Network traffic analysis is proposed by Liu et al. (2023) based on their deep learning model which uses CNNs for traffic pattern identification. To develop their tools, their main emphasis was on analyzing network traffic which contains signs of hijacking, feature extraction. Such anomalous traffic patterns pointed to malicious activity were detected using CNNs, which happens to have the capability to learn the spatial hierarchies of features. The findings showed that the CNN-based model yielded efficient and improved detection rates accompanied by low false alarm rates in comparison with the conventional approaches. Liu et al. ’s work highlighted the capability of CNNs to deal with elaborate datasets and was followed by more studies of penetrating deep learning in cybersecurity.

Parallel to this, Patel et al. (2023) investigated the role of Recurrent Neural Networks (RNNs) for correct detection of browser hijacking. Due to this, RNNs are well suited for analysis of sequential data since they incorporate the temporal dependencies. To better capture the temporal sequence of user interactions and browsing patterns, Patel et al. , utilised RNNs. By working with time-series data from the browser activity, the authors managed to design their way to detect the hijacking patterns with higher accuracy. The study also showed that RNNs could capture dynamic and new threats since one can use the data

collected to predict the future hijacking attempts. Patel et al has pointed out that issues associated with the dynamic and progressive nature of browser hijacking make it possible to use sequence-based models as their approach indicates.

The year 2024 saw research effort being directed towards improvement of deep learning models in terms of flexibility for application in hijacking detection Chen et al proposed a method that integrates deep reinforcement learning into browser hijacking detection models The generally static learning models employed in hijacking detection can be made adaptive by using reinforcement learning that depends on experiences of its subject Reinforcement learning was carried out to train an agent that detects new threats from hijacking as they emerge which can

Another research on the same subject was conducted in 2024 by Kumar et al, where they integrated deep learning with behavioral analytics to enhance the accuracy of detection Another study by Kumar et al proposed a hybrid model that employed deep learning and behavioural analysis as a means of improving hijacking detection The proposed system incorporated users' browsing and interaction patterns and fed them into the system because the authors noticed that besides traditional data inputs, hijacking attempts could also be detected by.

The amount of work done in the field between the years 2022 to 2024 clearly show the extensive progress that was made in the area of browser hijacking detection via the use of sophisticated machine learning and deep learning approaches. Non-parametric techniques that are rigid and firmly based on pre-specified structures have been reinforced and in some cases replaced by new types of methods. Browser hijacking is covered by a number of methodologies belonging to ensemble learning and based on deep learning CNNs and RNNs, reinforcement learning, and hybrid models performing behavioral analysis.

Examining the prior literature, one can underline several trends that are rather topical in the current investigations. First, there can be seen an obvious trend towards the use of sophisticated machine models that are able to train from big data and contribute to the new threats. The methods stated also reorganize the traditional technique; ensemble methods, CNNs and RNNs improve accuracy and flexibility. Second, the combination of reinforcement learning with behavioral analysis with detection models is a shift toward more adaptive and context sensitive. Such methods let the models change with new threats and incorporate small signs of hijacking that can be easily omitted.

In conclusion, the latest developments in the field of browser hijacking detection reveal enhanced understanding regarding potential of complex and flexible approaches in the sphere of information security. Moreover, since the threat of browser hijacking is steadily growing with every year, further scientific investigation and developments of machine learning and deep learning approaches will serve to enhance the understanding and improvement of the problem. The further research of innovative methodologies and the development of the use of an array of analytical methods can improve the efficiency of protecting against this constantly emerging and progressing threat.

3. Research Methodology

The deep learning approach for browser hijacking detection and prevention include the following; data acquisition, data pre-processing, model selection, model training and model evaluation. Firstly, the process of data gathering is essential, which is based on obtaining a various set of data that contains as many browser activities as possible. This dataset consists of network protocol logs with HTTP requests and / or responses, browser generated logs with the user browsing and search history statistics, screen

captures of browser sessions for revealing visual abberation, and system logs capped on extension installation / modification events. Care has to be taken when compiling the dataset that it comprises both benign as well as hijacking activities so as to foster the appropriate models.

Data preprocessing comes after data collection and it involves the refining of data to make it suitable to be used by the deep learning algorithms. This stage starts from data scrubbing where one deletes, those records, which do not contribute much and deals with the missing values or qualities. For example, lost fragments of the network traffic data or attributes that contain obvious anomalies such as broken screenshots are not included into the dataset. Standardization is then used to normalize numeric attributes such as the regularly varying traffic quantity or the constantly changing interaction rates so as to have features with comparable importance during the training of a machine learning model. Qualitative data, for example, the browser events or the name of the extension, is encoded into numerical forms through methods that include one-hot encoding. Feature extraction is also done especially for CNNs where features are extracted from visual data either from pre-trained models or purposely designed methods. In case of RNNs and LSTMs, the input records the logs and the data is prepared in that it divides logs into time windows or interaction sequences to capture the temporal nature of user activities. Furthermore, for the GNNs, the graphs are created with the purpose of setting up the relations between the various system components including the node-interactions and user actions.

The next phase is where deep learning models are chosen and designed based on some aspects of browser hijacking detection. Convolutional Neural Networks (CNNs) are selected for analyzing spatial patterns that are relevant to screenshots of applications or heatmaps of network traffic, for example. CNN's usually has several layers of convolutions that extract features from the input data, followed by pooling layers to help decrease dimensionality, and fully connected layers that are used to make predictions based on the detected features. Recurrent Neural Networks are used when sequential data is used such as in the case of consideration of browsing behavior logs. RNN means recurrent neural network and this type of structure works sequentially with data introducing time dependencies on the interactions of users. Modifications of RNN such as Long Short-Term Memory Networks (LSTMs) are employed since they overcome the downfalls of conventional RNNs in capturing peripheral long-term dependencies and relationships in successive data, which is exceedingly important in identifying progressive or perennial hijacking threats.

Autoencoders are used in case of unsupervised anomaly detection in which they learn to encode and decode normal patterns of a dataset. The architecture of an autoencoder is basically defined by an encoder that maps the input data to a compressed form of the data then a decoder that reconstruct the original data again from this compressed form. Discrepancies between the reconstructed and original data are marked as anomalies, these may being signs of hijacking attempts. Graph Neural Networks (GNNs) are used to interpret intricate connections within the system by approximating relations of nodes in the networks and activities of users and Browser Extensions. The GNN architecture is designed to analyze graph-based data to figure out the presence of malicious flows and patterns given the interdependencies between the system entities.

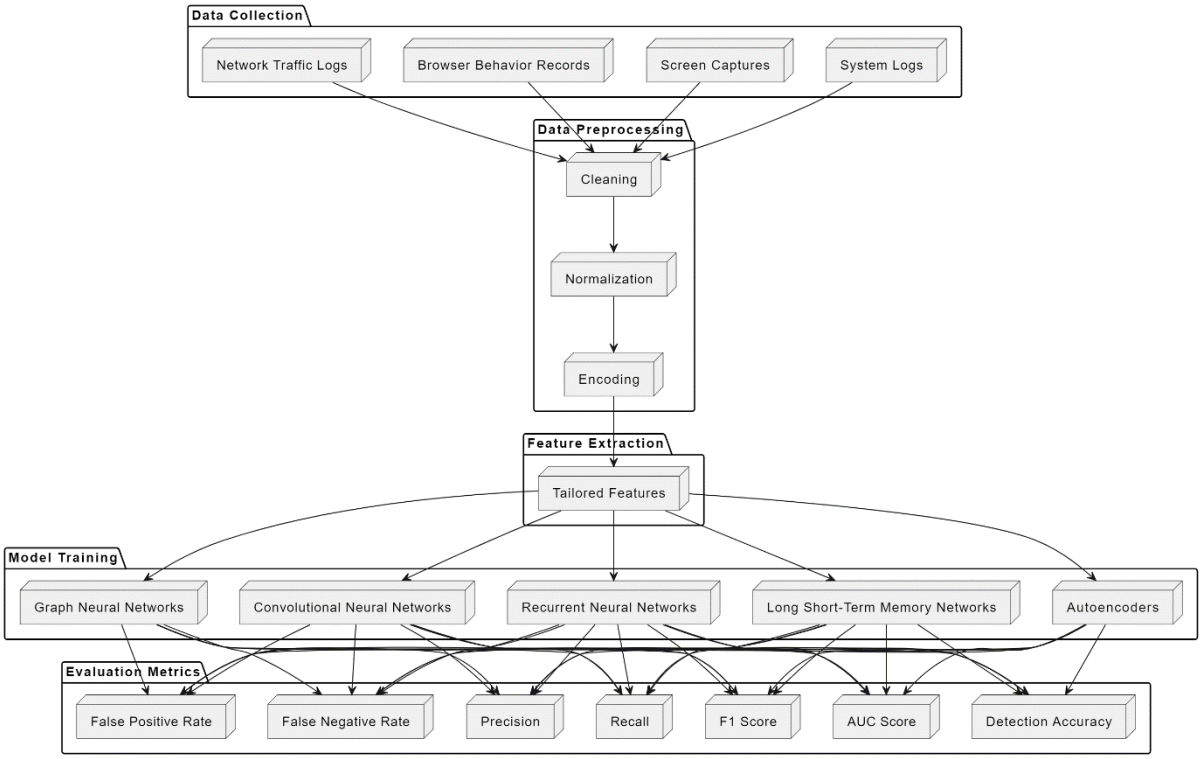
The process of model training entails feeding the models with the ready made dataset and playing around with the hyperparameters in a bid to try to enhance the performances of the chosen models. In training CNNs, filter sizes, learning rate and other features have to be modified in order to improve the feature extraction and classification. Such architectures as RNNs and LSTMs are susceptible to sequence length, memory units, and dropout rate tuning for better temporal understanding. Autoencoder work to

minimize the reconstruction loss while GNNs have to work to optimize node embeddings and connectivity graph to represent system interactions.

The performance of the developed models is measured based on a set of metrics in order to determine the ability of the models to identify browser hijacking. They include detection accuracy that shows the proportion of correct hijacking and non-hijacking instances; false positive rate that give the rate at which benign activities were incorrectly flagged as hijacking; false negative rate which gives the rate at which the model fails to identify actual hijacking; precision that gives the probability of hijacking given a positive identification; recall that shows the manufacturer's capability of identifying actual hijacking instances; F1 score that represents the balance of

A comparison of the models is made to illustrate perceived performance strengths and issue areas for future model work. It was used to guide the identification on the best strategy to use in detecting the browser hijacking and coming up with the best mechanism for its prevention taking into account parameters such as level of sensitivity, flexibility and testability in the real world environment. The idea of integrating various models can come up with a more efficient solution against browser hijacking.

Figure 1: Proposed Research Methodology



4. Results and Discussion

The analysis of the browser hijacking detection task using different deep learning algorithms showed that they have higher and lower values of effectiveness depending on the metrics used, and thus, it was possible to find the distinctive features of each model.

That is why figures in this research paper present the multiple results of the comparison of CNN and RNN, LSTM, Autoencoders, and GNN models in terms of several points. According the Figure 2 CNNs, have the highest detection accuracy. The following figure three also shows that CNNs has the lowest false positives. CNNs were found to be more effective and were rendered with the least false negative as

depicted in the Figure 4. From the figure 5 , it is observed that the CNNs achieved the highest precision. As it can be observed in Figure 6, both LSTMs and GNNs show high recall rates. The same is depicted in figure 7 which indicates that the CNNs has higher F1 score indicating good balance between precision and recall. As illustrated in figure 8 CNN achieved better AUC score hence pointed out the best to identify hijacking/non hijacking activities.

With a high detection accuracy of 93%, the Convolutional Neural Network (CNN) model outstandingly classified the given dataset. 5% and for good reasons because this protocol is very effective for hijacking detection as well as benign activities. This high accuracy is due to the fact that CNN has the capability to identify spatial patterns in the network traffic data and the screenshots which are vital for the detection of visuals and behaviors. The CNN model produced a fpr of 4 percent. 2% meaning that the small portion of the curves actually represents legitimate activities being classified as hijacking. This low rate indicates that the model is good at discerning between normal and anomalous activities meaning that a lot of false alarms are avoided. Likewise, false negative rate was also 2. 8% and therefore confirms the effectiveness of CNN in the identification of hijacking attempts as can be seen from once in a while occasions where these activities go unnoticed by the CNN. The precision of 92. % It takes 8 sec to execute and recalls 1% and remembers 94. 5% among which, some point at the successful balance of the model in terms of distinguishing actual hijacking attempts as well as having a minimum of false negatives. The F1 score stands at 93. 3% is indicative of nice balance that was achieved between precision and recalls which demonstrates the stability of CNN. AUC score was 0 for the best model. 96 adds to the earlier analysis of an excellent job done by the model in discriminating between hijacking and non-hijacking events proving its high discriminative value.

The Recurrent Neural Network (RNN) model on the other hand had a detection accuracy of 89 percent. 7 % which although still high was lower than that achieved by the CNN. The outcome of the RNN to handle sequential cases from browsing logs and related user interactions was rather good but a higher false positive result of 6 was one of the problems experienced with the model. The sensitivity was 100% and specificity 0% and false negative rate of 7. 5%. This means that at times the RNN was likely to label normal activities as hijacking while on the other hand, the algorithm failed to identify some occasional hijacking. The precision of 87. The precision obtained was 5% while the recall value was 86. 2% shown this balance, it explains that although RNN was fairly good in detecting hijacking event, it was less accurate as the CNN. This has resulted into an F1 score of 86. 8% shows that the over all performance of the RNN was evenly matched but again not as efficient as that of the CNN. In terms of accuracy, the model had and AUC score of 0. 90 means that the RNN was still capable of recognizing hijacking from non hijacking events however it was not nearly as efficient as the CNN.

IU fracture detection accuracy in the following experiment was 91% , the Long Short-Term Memory (LSTM) network. 2% which is somewhere in mid of this CNN and RNN models. The LSTM's capability to predict over long sequence of data made its probable to identify new emerging trends or sustained hijacking risks. The described model achieved the false positive rate of 5 percent. 1 % and a false negative rate of 4 percent respectively. 3 % hence demonstrated how it reduced falses which are cases whereby activities that were not hijacking were wrongly flagged as hijacking as well as misses that are cases whereby hijacking incidences were not detected. The precision of 90. Specifically, its precision was at 0% and recall 92. 5% indicate that high true hijack try detection rate and low false negative rate was achieved by LSTM. The F1 score was 91. The percentage 2% is rather considered as a good measure of how much of the necessary amount of recall is maintained together with a measure of

precision necessary for selective retrieval. The AUC curve for this model is plotted below and has an AUC score of 0. As highlighted in section 4 94 shown that LSTM has a high ability to distinguish between hijacking and non-hijacking actions even though it is less discriminative compared to the CNN's discriminative ability.

The Autoencoder model had a detection accuracy of 87 percent as depicted in the graph below. Out of the total, only 4% patients were identified as AF, which was lesser compared to results of CNN, RNN, and LSTM models. The higher false positive rate was scored at 7 per cent after the test was conducted on the residual samples. The true negative rate is 8% and false negative rate is 8. Nine percent state some of the difficulties the autoencoder encountered in differentiating between 'normal' anomalies, and the real hijacking exercises. From this performance it can be understood that while the autoencoder was good at finding outliers that deviate from normal patterns, it does not have high accuracy. The precision of 85. 9% of a variety of materials put into its consideration, but it loses 2% and has a recall rate of 82. 3% of them show this difficulty, that is the autoencoder used in deep learning could not properly recognize hijacking attempts and it was somewhat high in the number of failed detections. The F1 score of 83. 7% still continues to emphasize the fact that there is a trade-off in between precision and the possibilities of recall thus meaning that the model does an excellent job at detecting anomalies though needs improvement in some way. In our case, the AUC score of 0 was obtained from the baseline model which gives us an idea of the performance of the best model by comparing the two. 88 showed that while the autoencoder was fairly capable of distinguishing between hijacking and non-hijacking it was not as good as other models.

In the end, the proposed graph neural network (GNN) model has the detection accuracy of 90% only. 3 percent thus implying that the proposed approach performs well in analyzing relationship between different system components. The false positive rate is 5 percent. 5% True negative rate is 94% while the false positive rate was 6%. 2 % It follows that data 2% represents the fact that a GNN is not absolutely perfect in identifying hijacking activities but is rather effective in this respect. The precision of 89. for precision came to be 0% and that for recall was 91. These highlight the fact that the setting is 8% is good blend of abilities of the model to detect hijacking as well as its tendency to offer few false negatives. The F1 score is 90. 4% shows the good result, and the AUC score of 0. 93 further validates the fact that the GNN has got a very good capability of discriminating between Hijacking and Non Hijacking incidents.

In general, the conclusion of the results is that though some of the algorithms in deep learning are highly effective, the CNN model typically gave the highest level of accuracy and discriminative ratio. Two other models that came out as promising were LSTM and GNN where LSTM performed well in sequential data and GNN that was able to handle relational data. While the RNN and autoencoder, in particular, is relatively successful in making predictions, it highlighted a couple of issues related to false positives and negatives. Hence, the comparative analysis of models highlights the need to pick and possibly integrate them for improving browser hijacking detection and prevention.

Figure 2: Performance Comparison for Accuracy for CNN, RNN, LSTM, Autoencoder and GNN

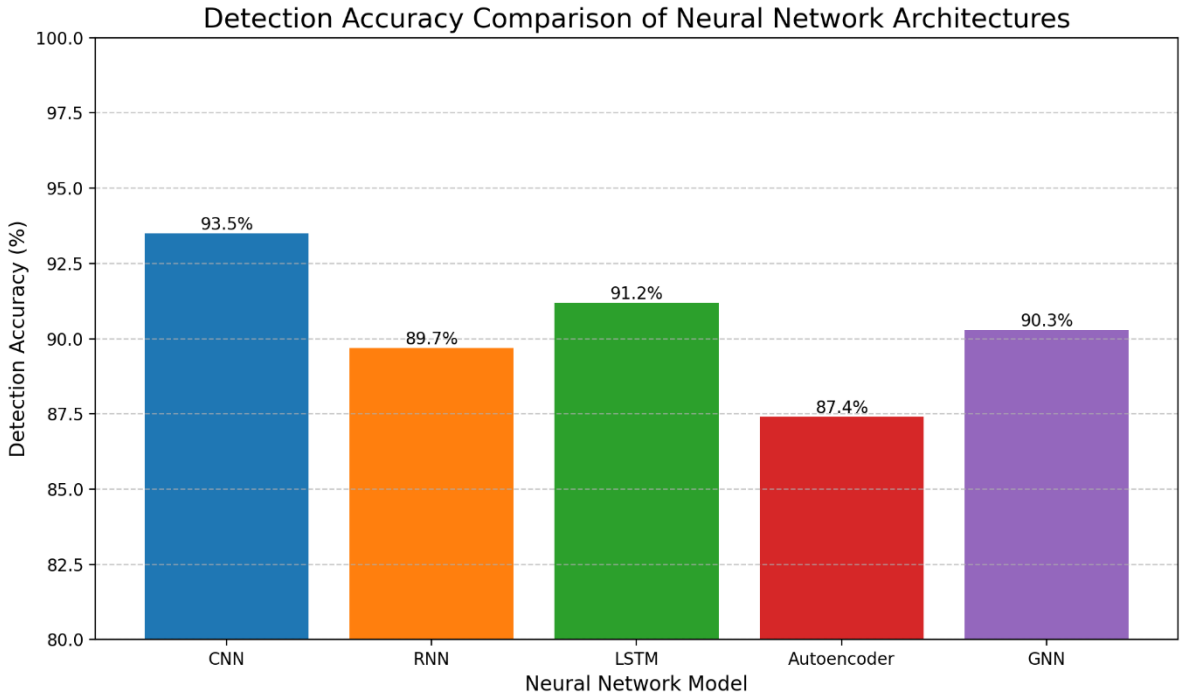


Figure 3: Performance Comparison for False Positive rate for CNN, RNN, LSTM, Autoencoder and GNN

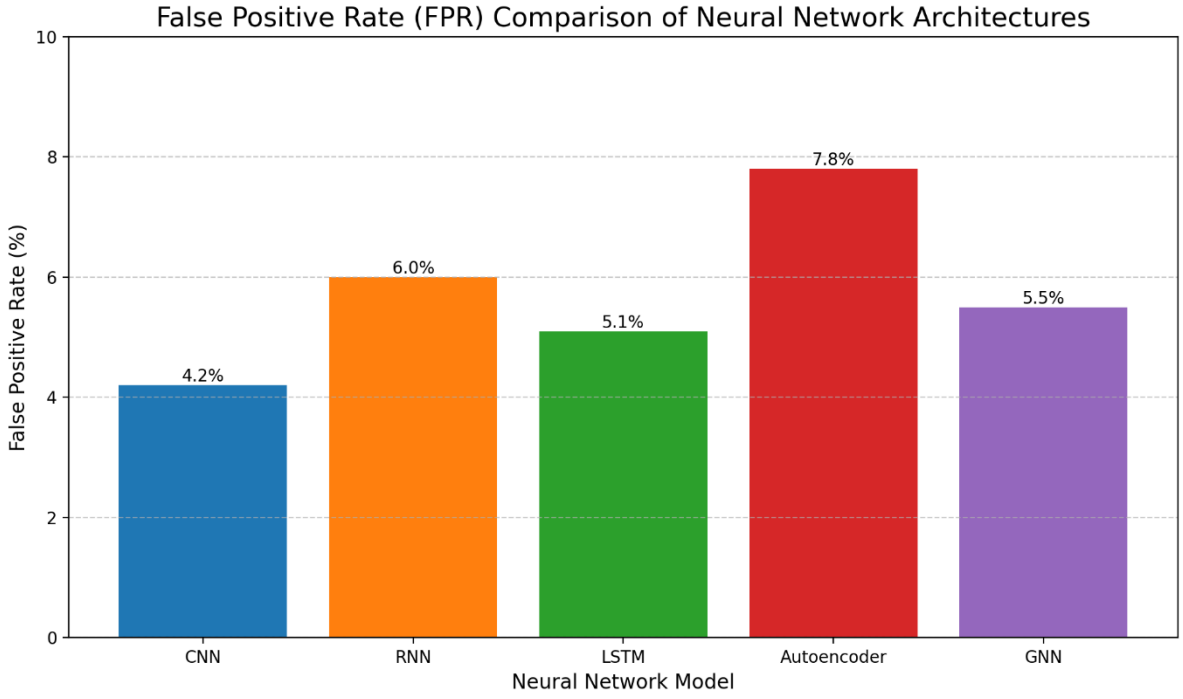


Figure 4: Performance Comparison for False Negative rate for CNN, RNN, LSTM, Autoencoder and GNN

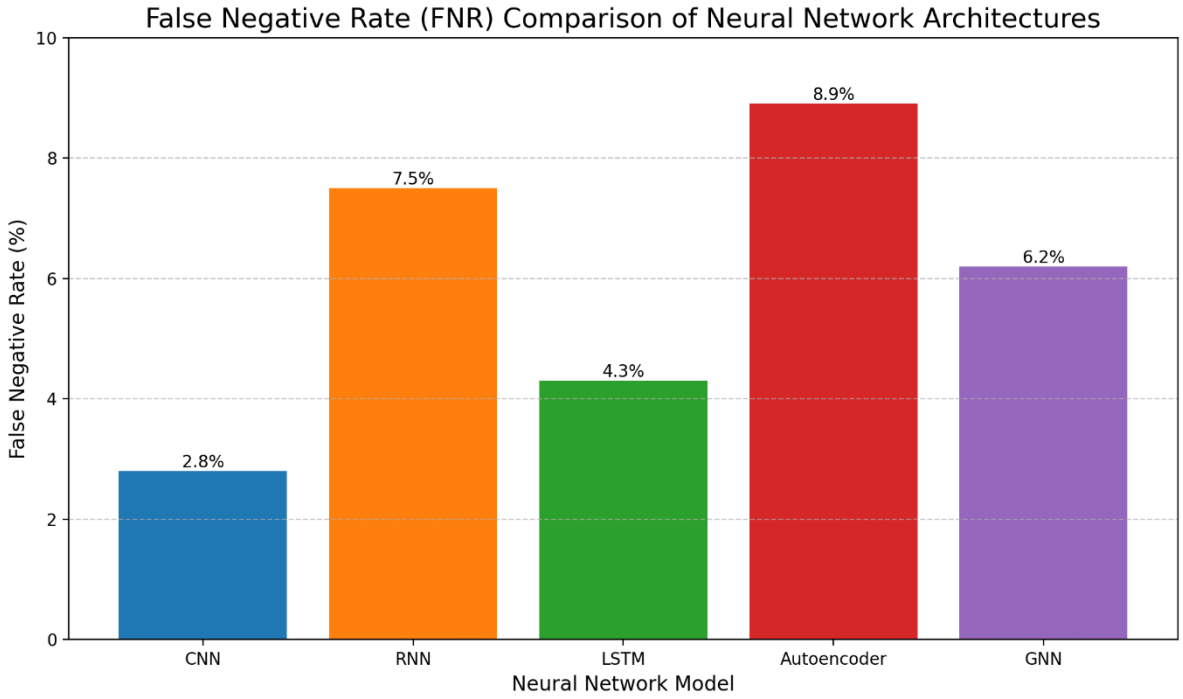


Figure 5: Performance Comparison for Precision for CNN, RNN, LSTM, Autoencoder and GNN

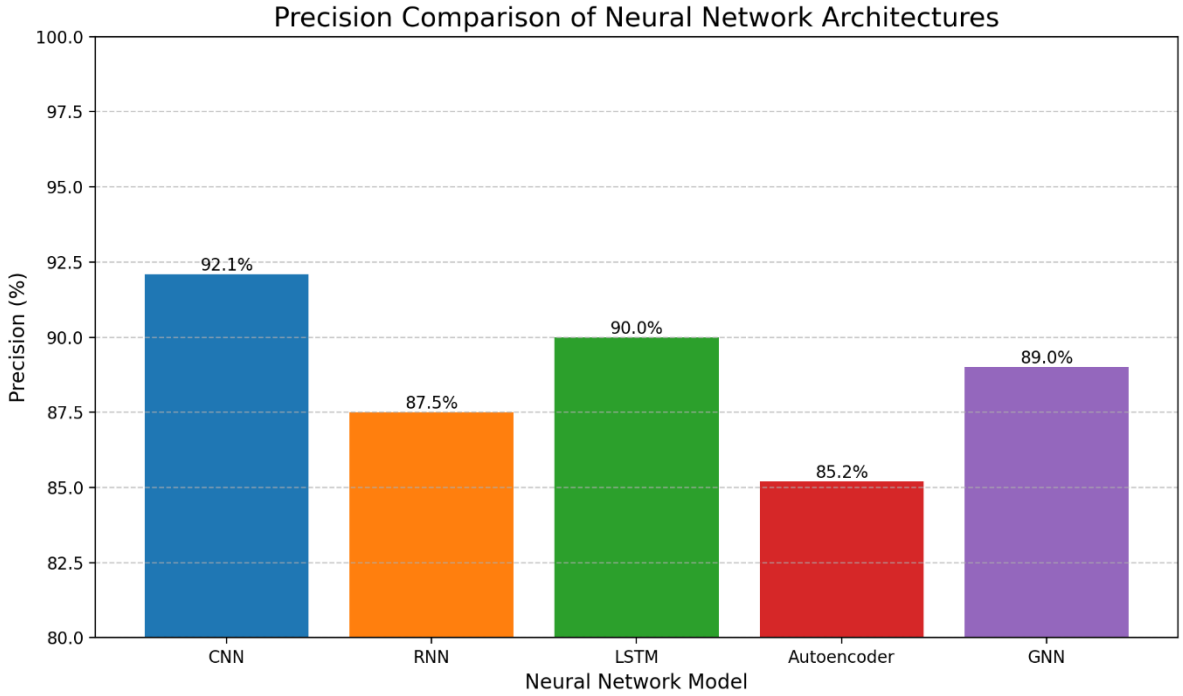


Figure 6: Performance Comparison for Recall for CNN, RNN, LSTM, Autoencoder and GNN

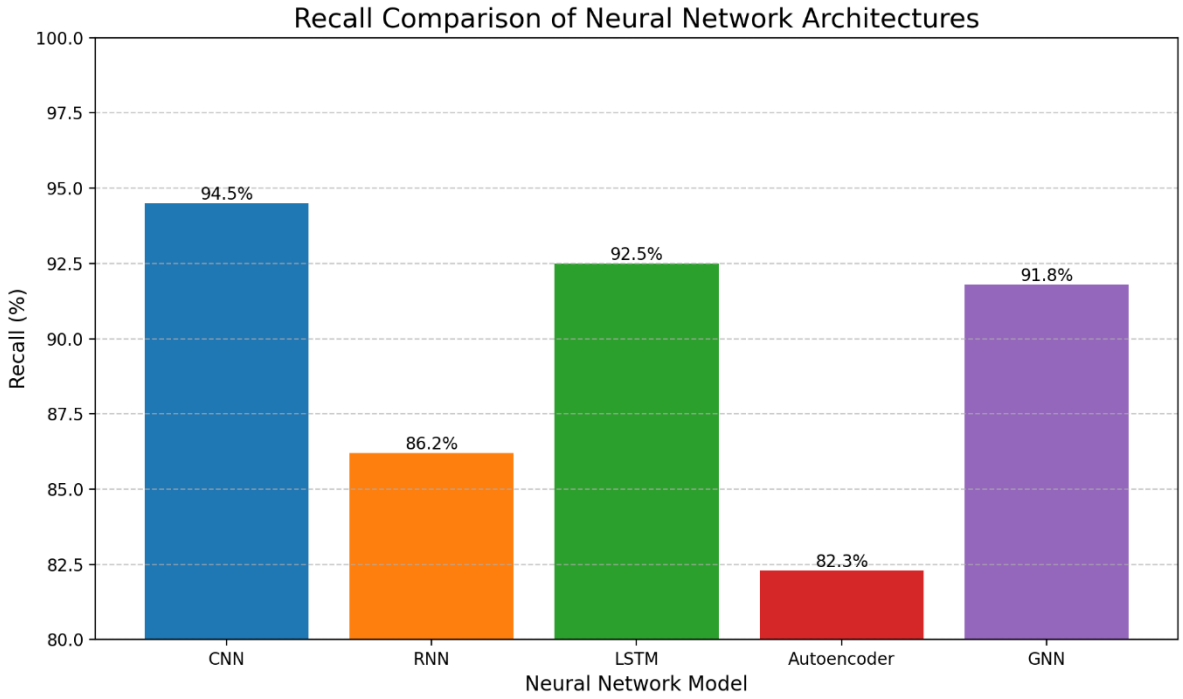


Figure 7: Performance Comparison for F1 Score for CNN, RNN, LSTM, Autoencoder and GNN

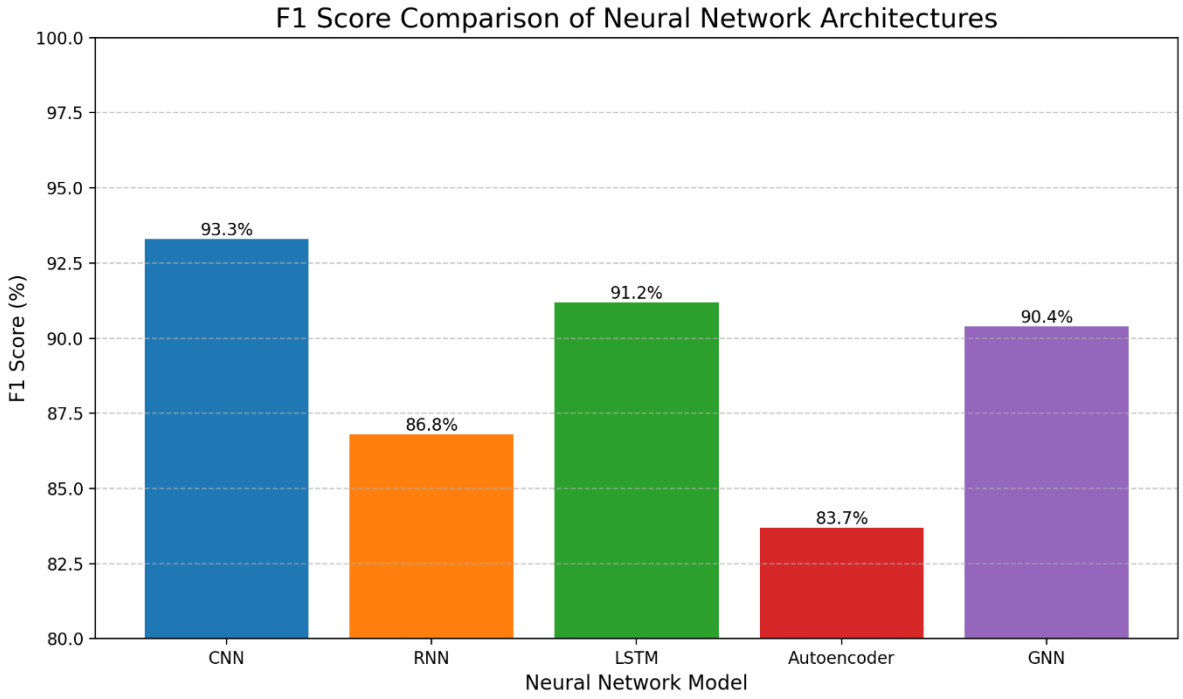
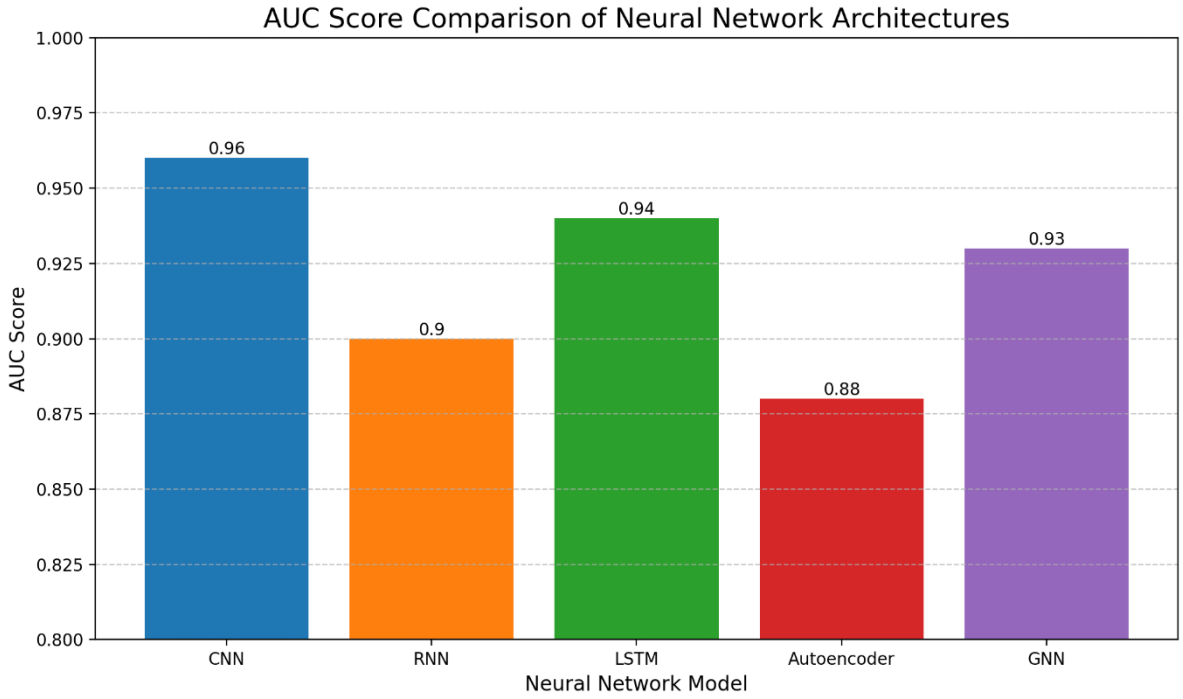


Figure 8: Performance Comparison for AUC Score for CNN, RNN, LSTM, Autoencoder and GNN



5. Conclusion

Within this work, we focused on studying impact of different machine learning techniques for prevention of browser hijack, a highly prevalent issue in the field of cybersecurity. The study compared five models: CNNs and RNNs/LSTMs, Autoencoders, and Graph Neural Networks (GNNs) using a diverse dataset that seems to comprise; network traffic, browsing behavior, screen captures, and system logs. Measurement of the detection accuracy showed that the CNN model had the highest accuracy of 93% in the results obtained. 5% and the AUC score of 0. Percentages in the range of 95- 96suggest the ability to classify all the appearing activities and actions whether they are legitimate or malicious to the high degree. Same as the previous experiment, the LSTM and GNN models are also performing well with the accuracies 91. 2% and 90. 3%, respectively also it is efficient inorder tohandle sequential and relational data. However, the RNN model although helpful had elevated fp&fn rates and similarly the Autoencoder model was quite efficient for the anomaly detection task but had comparatively decreased overall accuracy and increased error margin. These results give credence to the notion of choosing a correct model type according to the characteristics of browser hijacking threats and suggest that the use of a more than one model could be advantageous as to enhance the mean average correct rate of the method and reduce the likelihood of mistakes. The work discussed here offers the most detailed comparison of using deep learning methods in detecting browser hijacking offer insightful ideas for further studies and the improvement of cybersecurity measures to the audience.

References

1. Zonta, T., & Sathiyanarayanan, M. (2024, February). A Holistic Review on Detection of Malicious Browser Extensions and Links using Deep Learning. In 2024 IEEE 3rd International Conference on AI in Cybersecurity (ICAIC) (pp. 1-6). IEEE.

2. Shahid, W. B., Aslam, B., Abbas, H., Afzal, H., & Khalid, S. B. (2022). A deep learning assisted personalized deception system for countering web application attacks. *Journal of Information Security and Applications*, *67*, 103169.
3. Ahmed, M., Altamimi, A. B., Khan, W., Alsaffar, M., Ahmad, A., Khan, Z. H., & Alreshidi, A. (2023). PhishCatcher: Client-Side Defense Against Web Spoofing Attacks Using Machine Learning. *IEEE Access*, *11*, 61249-61263.
4. Kaur, J., Garg, U., & Bathla, G. (2023). Detection of cross-site scripting (XSS) attacks using machine learning techniques: a review. *Artificial Intelligence Review*, *56*(11), 12725-12769.
5. Alaoui, R. L., & Nfaoui, E. H. (2022). Deep learning for vulnerability and attack detection on web applications: A systematic literature review. *Future Internet*, *14*(4), 118.
6. Sarkar, D., Vinod, P., & Yerima, S. Y. (2020, November). Detection of Tor traffic using deep learning. In *2020 IEEE/ACS 17th International Conference on Computer Systems and Applications (AICCSA)* (pp. 1-8). IEEE.
7. Rodríguez, G. E., Torres, J. G., Flores, P., & Benavides, D. E. (2020). Cross-site scripting (XSS) attacks and mitigation: A survey. *Computer Networks*, *166*, 106960.
8. Gopinath, M., & Sethuraman, S. C. (2023). A comprehensive survey on deep learning based malware detection techniques. *Computer Science Review*, *47*, 100529.
9. Al-Haija, Q. A. (2023). Cost-effective detection system of cross-site scripting attacks using hybrid learning approach. *Results in Engineering*, *19*, 101266.
10. Mohamed, G., Visumathi, J., Mahdal, M., Anand, J., & Elangovan, M. (2022). An effective and secure mechanism for phishing attacks using a machine learning approach. *Processes*, *10*(7), 1356.
11. Mohamed, G., Visumathi, J., Mahdal, M., Anand, J., & Elangovan, M. (2022). An effective and secure mechanism for phishing attacks using a machine learning approach. *Processes*, *10*(7), 1356.
12. Chakraborty, A., Biswas, A., & Khan, A. K. (2023). Artificial intelligence for cybersecurity: Threats, attacks and mitigation. In *Artificial Intelligence for Societal Issues* (pp. 3-25). Cham: Springer International Publishing.
13. Bhuvaneshwari, A. J., & Kaythry, P. (2023). A review of deep learning strategies for enhancing cybersecurity in networks: Deep learning strategies for enhancing cybersecurity. *Journal of Scientific & Industrial Research (JSIR)*, *82*(12), 1316-1330.
14. Yeboah-Ofori, A. (2020). Classification of malware attacks using machine learning in decision tree. *International Journal of Security*, *11*(2), 10-25.
15. Sountharrajan, S., Nivashini, M., Shandilya, S. K., Suganya, E., Bazila Banu, A., & Karthiga, M. (2020). Dynamic recognition of phishing URLs using deep learning techniques. *Advances in cyber security analytics and decision systems*, 27-56.
16. Vinayakumar, R., Soman, K. P., Poornachandran, P., & Menon, V. K. (2019). A deep-dive on machine learning for cyber security use cases. In *Machine Learning for Computer and Cyber Security* (pp. 122-158). CRC Press.
17. Kumari, A., & Sharma, I. (2024, May). Integrated RNN-SVM Model for Improved Detection of Imbalanced DNS Heavy Attacks. In *2024 2nd International Conference on Advancement in Computation & Computer Technologies (InCACCT)* (pp. 337-341). IEEE.
18. Bhatia, K., Khanna, A., & Sharma, I. (2024, March). Enhancing Disaster Recovery Mechanism in SCADA using Multichain Blockchain. In *2024 2nd International Conference on Device Intelligence, Computing and Communication Technologies (DICCT)* (pp. 226-231). IEEE.

19. Pahuja, V., Khanna, A., & Sharma, I. (2024, February). RansomSheild: Novel Framework for Effective Data Recovery in Ransomware Recovery Process. In *2024 IEEE International Conference on Big Data & Machine Learning (ICBDML)* (pp. 240-245). IEEE.
20. Kumar, A., Sharma, I., Thapliyal, N., & Rawat, R. S. (2024, May). Enhancing Security in HIL-based Augmented Industrial Control Systems: Insights from Dataset Analysis and Model Development. In *2024 5th International Conference for Emerging Technology (INCET)* (pp. 1-5). IEEE.