1. László GÖCS, 1.Zsolt Csaba JOHANYÁK

CATBOOST ALGORITHM BASED CLASSIFIER MODULE FOR BRUTE FORCE ATTACK DETECTION

1. John von Neumann University, GAMF Faculty of Engineering and Computer Science, Department of Information Technology, Kecskemét, HUNGARY

Abstract: Intrusion Detection Systems (IDS) play a critical role in safeguarding corporate IT systems by providing automated protection against various attacks and intrusions. They efficiently identify suspicious attack scenarios and promptly alert or intervene to prevent an attack. This paper focuses on the research conducted to develop a classification module for Behavior-based IDSs (BIDS). Typically, these modules are built using a sample dataset that comprises a significant amount of data describing both benign and malicious network traffic. In our case, the CSE-CIC-IDS2018 dataset on AWS served as the foundation for this purpose. The investigation aimed to develop an effective classifier module for a BIDS system using the CatBoost algorithm. To evaluate the performance of the trained classifier, we compared it to three other well-known classifiers trained on the same data, employing the same selected features. Additionally, we utilized four different performance measures, namely accuracy, precision, recall, and F1 score. The results demonstrated that, overall, the CatBoost classifier delivered performance on par with or better than the baseline methods. This finding supports the initial assumption that a CatBoost-based solution could be a viable choice when developing a BIDS.
Keywords: IDS, feature selection, classification, CatBoost

1. INTRODUCTION

Intrusion Detection Systems (IDSs) play a crucial role in protecting corporate IT systems by providing automated defense against various attacks and intrusions. IDSs actively search for specific events and traces across network or computer resources, which can indicate potential malicious activities or attacks. Once suspicious situations are detected, IDSs generate alerts or implement preventive measures to prevent an attack [1].

The results presented in this paper are based on an investigation focusing on anomaly-based intrusion detection systems, specifically Behavior IDSs (BIDSs). These systems operate in two modes: training and detection. In the training mode, the system is fed with sensor data that includes both typical (normal) network behavior and malicious (attack) data. The classification module is then trained and evaluated using labeled data records. In the detection mode, the fully trained classifier is deployed to assess whether the ongoing activity poses a threat to the system or not.

Feature selection and dimensionality reduction play a significant role in the training process of behavior-based IDSs. These techniques aim to identify the most relevant and informative features from the dataset while reducing its complexity. By selecting the right set of features and reducing dimensionality, the system can focus on the most important aspects of network traffic and effectively distinguish between normal behavior and potential intrusions. This optimization is particularly important in real-time monitoring scenarios, where timely detection and response to network threats are critical for maintaining the security and integrity of a network infrastructure. Therefore, feature selection and dimensionality reduction contribute to enhancing the overall performance and accuracy of behavior-based IDSs, making them essential steps in the training process.

In our previous research, we performed dimensionality reduction on the CSE-CIC-IDS2018 on AWS dataset [2] and employed six different feature selection methods [3]. The feature scores were normalized and aggregated using a weighted average approach, and the selected feature groups were tested with five different classification methods. The performance of each classifier was evaluated using four metrics.

The main assumption of the current investigation presented in this paper was that utilizing the CatBoost algorithm for classification could yield classification results that are at least as good as those obtained through previously investigated baseline classifier types.

The rest of this paper is organized as follows. Section 2 provides a brief overview of related research work. Section 3 presents the proposed CatBoost algorithm and the performance evaluation measures employed. Section 4 describes the experimental results, while the conclusions are presented in Section 5.

2. RELATED WORKS

In their research, Shiladitya Raj et al. performed machine learning test on NSLKDD dataset using CatBoost algorithm and compared it with ELM. The results show that CatBoost is a more efficient solution [4].

In his work, Viknesh Aditya Rajendran conducted a test on the IoT-BDA dataset using a joint solution that combines the LightGBM and CatBoost algorithms. The results of the test showcased an impressive accuracy rate of 99 percent and a low false alarm rate of 0.5 percent for the identification of normal and malicious data packets [5].

In their research, Alina Lazar and Alex Sim examined the effectiveness of utilizing GPUs by testing multiple classifiers (Multinomial Naive Bayes, K-Nearest Neighbor, Random Forest, Support Vector Machine, XGBoost, and CatBoost) on the AWID dataset. The findings demonstrated notable speedups, with improvements of up to 65 times observed [6].

Li Yang et al. employed an ensemble of XGBoost, LightGBM, and CatBoost machine learning algorithms, referred to as LCCDE (Leader Class and Confidence Decision Ensemble) on the Car-Hacking and CICIDS2017 datasets. The obtained results showed significant improvements compared to previous machine learning methods [7].

Latha R. et al. the Hybrid CatBoost algorithm was compared with Concolutional Neural Network (CNN), Deep CNN, and RSA using the IDS2017 dataset. The results showed an accuracy of 92.5% and a remarkable 99% reduction in the False Negative Rate (FNR) [8].

Sakshi Pandey et al. studies have demonstrated that CatBoost significantly improved DDoS attack detection to an impressive rate of 99.82%. Furthermore, CatBoost has outperformed other methods of various DDoS attacks. Notably, it achieved a false alarm rate as low as 0.4% [9].

3. METHODOLOGY

CatBoost algorithm

CatBoost (Category Boosting) is a special decision tree algorithm that belongs to the family of gradient boosting algorithms. Its key idea is to create an efficient model by building an ensemble of weak predictor decision trees. These trees are combined iteratively in each step using the residuals of the previous step. Thus the classification performance of the model improves gradually step-by-step. In the case of binary classification CatBoost uses the log loss as loss function but a custom loss function can also be employed. The final class corresponding to the current input is determined by a combination of the predictions given by the individual trees belonging to the ensemble. The main steps of the training of a CatBoost classifier that are presented below.

1. *Initialize* the model by the logarithm of the proportion of frequencies of the different classes (values of the target variable) of the training data sample.
2. *Calculate the gradients of the loss function* in function of the class predictions of the current model.
3. *Order categorical features* based on then statistical effects of the individual values on the target variable.
4. *Create decision trees* using a modified version of the gradient boosting algorithm. In each iteration a new tree is created by recursively partitioning the training data sample.
5. *Prevent overfitting* using L2 and Newton's method type regularization on the leaf values of the trees.
6. *Make class prediction* using a weighted voting technique that combines the results provided by the different trees of the ensemble.
7. *Calculate loss function.*
8. *Stop* if the performance of the model did not improve on the validation dataset for a certain number of iterations.
9. Go to step 2 [10]

CatBoost implements an adaptive approach to automatically adjust the model parameters during the iterative training process. This adaptive nature contributes to the algorithm's effectiveness. It offers several advantages, including its ability to achieve high accuracy in modeling, as well as it exhibits robustness to data noise and reduced sensitivity to overfitting, which can be problematic in the case of other algorithms. It supports multi-class classification as well and enables the evaluation of feature importance [11].

Performance Evaluation

The objective of binary classification is to train a classifier algorithm using a dataset that can accurately classify new, unknown examples. Each example in the training and test dataset is labeled with a class label that specifies its correct class. When evaluating a classifier, the usual measures can be categorized into two main groups, the basic measures (n_{TP}, n_{FP}, n_{TN}, and n_{FN}) and the derived measures (e.g. Accuracy, Precision, Recall, and F1 score). The meaning of the TP, FP, TN, FN abbreviations is explained below and summarized in Table 1. Here 0 indicates a benign traffic while 1 stands for an attack.

TP (True Positive): This happens when the classifier correctly identifies a data instance (attack) as positive, and the prediction is accurate (true attack).

FP (False Positive): This occurs when the classifier incorrectly identifies or the testing algorithm mistakenly categorizes examples belonging to the negative class as positive. Consequently, the result is positive, but the true class is negative (identified as an attack when it was not).

TN (True Negative): This takes place when the classifier correctly identifies or the testing algorithm accurately diagnoses examples belonging to the negative class. Thus, the result is negative, and the true class is also negative (no attack occurred, and the classifier correctly classified it as such).

FN (False Negative): This arises when the classifier incorrectly identifies or the testing algorithm mistakenly categorizes examples belonging to the positive class as negative. Consequently, the result is negative, but the true class is positive (an attack occurred, but the classifier failed to recognize it).

Table 1. Classification result types in the case of binary classification

Actual value	Predicted value	Result
0	0	TN
0	1	FP
1	0	FN
1	1	TP

The most used four derived performance metrics are Accuracy, Precision, Recall, and F1-score. These metrics can be calculated using the following formulas:

$$\mathrm{Accuracy} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FN}+\mathrm{FP}}, \tag{1}$$

$$\mathrm{Pr\,ecision} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \tag{2}$$

$$\mathrm{Re\,call} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \tag{3}$$

$$\mathrm{F1} = \frac{2(\mathrm{Pr\,ecision}\cdot\mathrm{Re\,call})}{\mathrm{Pr\,ecision}+\mathrm{Re\,call}}. \tag{4}$$

Training the Classifier

The CatBoost classification algorithm was trained and tested using the provided feature data. The implementation was carried out in Python programming language with the relevant modules. The execution within a Python environment yielded efficient and prompt results. The following section shows a general outline to train a CatBoost classifier:

```
from catboost import CatBoostClassifier
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Loading data
X = data.data  # Features
y = data.target  # Labels

# Trainig and test data splitting
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Initialising CatBoostClassifier
model = CatBoostClassifier(iterations=100, learning_rate=0.1, depth=6)

# Training the model
model.fit(X_train, y_train)

# Making predictions
```

```
y_pred = model.predict(X_test)

#Evaluation of accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

This code shows a simple example of how to use CatBoostClassifier. The dataset is split into training and test data with '*train_test_split()*'. The CatBoostClassifier class is initialized with the parameters '*iterations*', '*learning_rate*' and '*depth*'. The '*iterations*' parameter specifies the number of iterations, *learning_rate* the training rate, and '*depth*' the depth of the tree. The '*fit()*' function training the model on the training data, and then '*predict()*' is used to make predictions on the test data.

4. EXPERIMENTAL RESULTS

In our previous research [2] [3], our aim was to create binary classifiers for five brute force attack types, namely FTP, SSH, Web, XSS, and SQL. For each of them separate training and test datasets were created from the original CSE-CIC-IDS2018 dataset, each dataset containing only one attack type and benign traffic data. All features were normalized to the [0,1] interval. Next, we conducted dimension reduction on the CSE-CIC-IDS 2018 on AWS dataset. Subsequently, we utilized various feature selection methods, including Information Gain, Gain Ratio, Relief, Anova, Symmetric Uncertainty, and Chi-square [2], to determine the relevant features.

Dimensionality reduction offers several significant advantages. One primary benefit is that many data mining algorithms exhibit improved performance when dealing with a smaller number of dimensions, which corresponds to a reduced number of attributes (columns) in the data. This improvement is partly due to the elimination of irrelevant attributes and the reduction of noise. Additionally, dimensionality reduction can contribute to a more comprehensible model by reducing the number of attributes it encompasses.

The resulting scores were normalized and aggregated by calculating a weighted average of the scores. To optimize the weights, we employed a Taguchi Design of Experiments (DoE) approach [3]. This approach facilitated the reduction of the number of features, leading to improved results.

The selected features were tested using different classification methods, such as Random Forest, Decision Tree, Naïve Bayes, Logistic Regression, and SVM. Finally, we evaluated the performance of each classifier using the Accuracy, Precision, Recall, and F1-score metrics on both the training and test datasets.

In course of the current research in the case of all attack types we used with the CatBoost classifier the same train and test data sets as well as features that were previously identified using the weighted average aggregation method of the individual feature scores [3].

In the case of the FTP attack type the resulting number of right and misclassified cases (n_{TP}, n_{FP}, n_{TN}, and n_{FN}) are summarized in Tables 2 and 3 for the train and test datasets, respectively.

The results supported the original assumptions regarding the applicability of the CatBoost based classifier for intrusion detection purposes especially in the case of the FTP and SSH attack types. For example, the resulting classifier was almost perfect in the case of the FTP attack. One misclassified sample in such big data set is acceptable. In general, the performance measures are satisfactory and if one takes into consideration that the time necessary for the training of the classifier was about one tenth of the time necessary to train the baseline classifiers the CatBoost based solution becomes even more attractive.

Table 2. Confusion Matrix for the FTP train dataset

n= 171.433	Predicted: 1	Predicted: 0	
Actual: 1	132.761	1	132.762
Actual: 0	0	38.671	38.671
	132.761	38.672	

Table 3. Confusion Matrix for the FTP test dataset

n= 85.716	Predicted: 1	Predicted: 0	
Actual: 1	66.380	1	66.381
Actual: 0	0	19.335	19.335
	66.380	19.336	

Similar to the previous investigation, separate binary classifiers were trained for each attack type and they were evaluated against the training and test datasets by Accuracy, Precision, Recall, and F1-score measures. The results are shown in Table 4.

Table 4. CatBoost results on datasets

Dataset		Accuracy	Precision	Recall	F1
FTP	training	1.0000	1.0000	1.0000	1.0000
	testing	1.0000	0.9999	1.0000	0.9997
SSH	training	1.0000	0.9999	1.0000	1.0000
	testing	1.0000	0.9998	1.0000	0.9999
WEB	training	0.9996	0.9978	0.7463	0.8539
	testing	0.9993	0.9978	0.7463	0.8539
XSS	training	0.9998	0.7926	0.9304	0.8560
	testing	0.9998	0.9264	0.9304	0.9224
SQL	training	1.0000	0.9759	0.9310	0.9529
	testing	1.0000	1.0000	0.9310	0.9643

5. CONCLUSIONS

The efficiency of network intrusion detection systems depends to a significant degree on the underlying classification module that determined from the current traffic data whether the system is under attack or not. The investigation reported in this paper focused on the applicability of the CatBoost algorithm for the role of classification module.

Therefore, in the case of five attack types (i.e., FTP, SSH, Web, XSS, and SQL) binary CatBoost classifiers were trained and tested using the feature sets defined by a weighted ensemble feature selection method [3]. The study revealed that in most of the cases employing the CatBoost algorithm resulted in at least the same of sometimes better classification performance for the investigated brute force attacks compared to the previously tested baseline classifier types.

Although the CatBoost classification performance measures were not always better but a big advantage of CatBoost was its significantly faster training time, which was on average, one-tenth that of the baseline classifiers.. Thus it can be said that the CatBoost algorithm can be an efficient and fast way to develop and operate anomaly-based IDS systems.

Our future research will focus on the investigation of the of the applicability of different fuzzy methods (e.g. [12], [13], [14], [15]) in the feature selection process.

Acknowledgements

This research was supported by 2020-1.1.2-PIACI-KFI-2020-00062 "Development of an industrial 4.0 modular industrial packaging machine with integrated data analysis and optimization based on artificial intelligence error analysis". The Hungarian Government supports the Project and is co-financed by the European Social Fund.

References

[1] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, 'Intrusion detection system: A comprehensive review', Journal of Network and Computer Applications, vol. 36, no. 1, pp. 16–24, Jan. 2013

[2] L. Göcs and Z. C. Johanyák, 'Identifying Relevant Features of CSE-CIC-IDS2018 Dataset for the Development of an Intrusion Detection System', vol. under publication, 2023.

[3] L. Göcs and Z. C. Johanyák, 'Feature Selection with Weighted Ensemble Ranking for Improved Classification Performance on the CSE-CIC-IDS2018 Dataset', Computer Science and Mathematics, preprint, Jul. 2023

[4] M.Tech, Department of Computer Science, Lakshmi Narain College of Technology Excellence Bhopal (M.P.), India, S. Raj, M. Jain, Assistant Professor, Department of Computer Science, Lakshmi Narain College of Technology Excellence Bhopal (M.P.), India., Dr. P. Chouksey, and Professor, Department of Computer Science, Lakshmi Narain College of Technology Excellence Bhopal (M.P.), India., 'A Network Intrusion Detection System Based on Categorical Boosting Technique using NSL-KDD', IJCNS, vol. 1, no. 2, pp. 1–4, Nov. 2021

[5] V. A. Rajendran, 'Ensemble Techniques to Enhance Wireless Intrusion Detection System In IoT'.

[6] A. Lazar, A. Sim, and K. Wu, 'GPU-based Classification for Wireless Intrusion Detection', in Proceedings of the 2021 on Systems and Network Telemetry and Analytics, Virtual Event Sweden: ACM, Jun. 2020, pp. 27–31

[7] L. Yang, A. Shami, G. Stevens, and S. De Rusett, 'LCCDE: A Decision-Based Ensemble Framework for Intrusion Detection in The Internet of Vehicles', in GLOBECOM 2022 - 2022 IEEE Global Communications Conference, Rio de Janeiro, Brazil: IEEE, Dec. 2022, pp. 3545–3550.

[8] R. Latha and R. M. Bommi, 'Hybrid CatBoost Regression model based Intrusion Detection System in IoT-Enabled Networks', in 2023 9th International Conference on Electrical Energy Systems (ICEES), Chennai, India: IEEE, Mar. 2023, pp. 264–269

[9] S. Pandey and S. Lahoti, 'Intrusion Detection System Analysis Using Cat boost Technique Intended for Cloud Communication Network', in 2023 IEEE International Conference on Integrated Circuits and Communication Systems (ICICACS), Raichur, India: IEEE, Feb. 2023, pp. 1–5

[10] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, 'CatBoost: unbiased boosting with categorical features'.

[11] S. R, A. Raj, K. Saivenu, M. I. Ahmed, S. B, and A. Kanavalli, 'Detection and mitigation of botnet based DDoS attacks using catboost machine learning algorithm in SDN environment', IJATEE, vol. 8, no. 76, pp. 445–461, Mar. 2021
[12] I.-D. Borlea, R.-E. Precup, A.-B. Borlea, and D. Iercan, 'A Unified Form of Fuzzy C-Means and K-Means algorithms and its Partitional Implementation', Knowledge-Based Systems, vol. 214, p. 106731, Feb. 2021
[13] S. Blazic and I. Skrjanc, 'Incremental Fuzzy C-Regression Clustering From Streaming Data for Local-Model-Network Identification', IEEE Trans. Fuzzy Syst., vol. 28, no. 4, pp. 758–767, Apr. 20v20
[14] J. Hvizdos, J. Vascak, and A. Brezina, 'Object identification and localization by smart floors', in 2015 IEEE 19th International Conference on Intelligent Engineering Systems (INES), Bratislava, Slovakia: IEEE, Sep. 2015, pp. 113–117.
[15] D. Vincze, A. Toth, and M. Niitsuma, 'Antecedent Redundancy Exploitation in Fuzzy Rule Interpolation-based Reinforcement Learning', in 2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Boston, MA, USA: IEEE, Jul. 2020, pp. 1316–1321.

ISSN 1584 – 2665 (printed version); ISSN 2601 – 2332 (online); ISSN–L 1584 – 2665