arXiv:2408.16034v1 [cs.CR] 28 Aug 2024

Systematic Evaluation of Synthetic Data Augmentation for Multi-class NetFlow Traffic ⋆

Maximilian Wolf[1] Dieter Landes[1] Andreas Hotho[2] Daniel Schlör[2]

[1] Center for Responsible Artificial Intelligence, University of Applied Sciences and Arts Coburg, Friedrich-Streib-Str. 2 Coburg, Germany {maximilian.wolf, dieter.landes}@hs-coburg.de

[2] Center for Artificial Intelligence and Data Science, University of Würzburg, Campus Hubland Nord, Emil-Fischer-Straße 50 Würzburg, Germany {hotho, schloer}@informatik.uni-wuerzburg.de

Abstract. The detection of cyber-attacks in computer networks is a crucial and ongoing research challenge. Machine learning-based attack classification offers a promising solution, as these models can be continuously updated with new data, enhancing the effectiveness of network intrusion detection systems (NIDS). Unlike binary classification models that simply indicate the presence of an attack, multi-class models can identify specific types of attacks, allowing for more targeted and effective incident responses. However, a significant drawback of these classification models is their sensitivity to imbalanced training data.

Recent advances suggest that generative models can assist in data augmentation, claiming to offer superior solutions for imbalanced datasets. Classical balancing methods, although less novel, also provide potential remedies for this issue. Despite these claims, a comprehensive comparison of these methods within the NIDS domain is lacking. Most existing studies focus narrowly on individual methods, making it difficult to compare results due to varying experimental setups. To close this gap, we designed a systematic framework to compare classical and generative resampling methods for class balancing across multiple popular classification models in the NIDS domain, evaluated on several NIDS benchmark datasets.

Our experiments indicate that resampling methods for balancing training data do not reliably improve classification performance. Although some instances show performance improvements, the majority of results indicate decreased performance, with no consistent trend in favor of a specific resampling technique enhancing a particular classifier.

Keywords: Imbalanced learning · Intrusion detection · Attack classification · Generative model.

⋆ Preprint; Accepted at Machine Learning for CyberSecurity @ European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD) 2024

1 Introduction

Cybercrime and its detection and prevention is an ongoing topic of research. A primary focus of attacks is the computer infrastructure of companies, which can be targeted in various ways. The ENISA report of 2023 [8] identifies key threats such as data theft, ransomware, and threats to system availability, including denial-of-service attacks and malware. To detect network attacks, network intrusion detection systems (NIDS) are used to analyze a company's network traffic and identify malicious activities. Previous studies have demonstrated the application of machine learning methods for the analysis and extraction of network traffic patterns, as highlighted in the comprehensive review by Salman et al. [34].

In this work, we focus on network intrusion detection as a multi-class classification problem, where the classifier must distinguish between different types of attacks. In contrast to binary classification approaches, which only indicate whether an attack has occurred, a multi-class approach provides specific information about the type of attack, allowing for a tailored response to the specific threat, such as port scans, brute-force attacks or DDoS. Since attack events are specific, but rare, their recording for benchmark purposes results in unbalanced class distributions. However, many standard multi-class models, such as Decision Trees, are sensitive to the class distribution and deliver low classification performance when trained on imbalanced datasets in other data domains [18]. Consequently, multiple ways of handling imbalanced class distributions have been proposed, e.g. classification algorithms specifically for imbalanced learning [19], weighting schemes [38], cost-sensitive learning [5] or resampling [10], which is particularly versatile as it is model-agnostic.

Resampling techniques, specifically under- and oversampling, balance the class distribution by drawing samples from the majority classes to match the number of entries in the minority class or vice versa. To further augment the training data, synthetic resampling methods such as SMOTE [3] balance the dataset by generating synthetic data. These common approaches might introduce issues in high-dimensional and complex datasets, such as boundary bias [9] and issues related to data-quality [12,2] as a linear feature interpolation may not generate valid data points that accurately represent real-world scenarios, potentially leading to reduced classifier performance. In contrast, conditional generative models such as Variational Autoencoders [23], Generative Adversarial Networks [13] and diffusion-based models such as Latent-Diffusion [33] promise to generate realistic synthetic data [32,6], applicable for data augmentation in terms of class balancing. In particular, some works claim that the tested generative models outperform classical oversampling techniques in terms of augmentation performance [16,26]. In contrast, one work on systematic evaluation of classical resampling techniques in the NetFlow domain [17] reports mixed results and emphasizes that none of the tested techniques reliably improves classification performance and can even decrease the performance of classifiers.

In the domain of NIDS, existing studies either evaluate classical over- and undersampling methods without including modern generative models for data augmentation [17], or they conduct individual comparisons of generative models

focusing solely on augmentation without incorporating classical sampling and cleaning techniques [25]. Moreover, the lack of standardization leads to individual generative models being tested on different datasets, each with distinct NetFlow features, which complicates direct comparisons of data generation approaches. Often, approaches are evaluated on a single dataset, raising questions about the generalizability of the results to other datasets. Furthermore, the arbitrarily chosen classifier models used to evaluate the resampling methods further complicates the overall comparability.

To address these gaps, our objective is to systematically evaluate the effectiveness of modern generative models to improve class balance through data augmentation. We compare these methods with classical (synthetic) oversampling techniques, both independently and in combination with undersampling methods. Specifically, we construct a testbed of various multi-class classifier models, which are evaluated using established classification metrics on NIDS benchmark datasets. This setup allows us to analyze the effects of both classical and modern resampling methods on different classification models. An overview of our experimental setup is given in Figure 1.

Our experiments generally reveal a minor negative impact of most resampling strategies on overall performance and rare classes in particular, while specific classifier-dependent combinations show improvements in classification performance. This suggests that the arbitrary choice of resampling and augmentation techniques is not a reliable method for improving multi-class classification performance on NIDS data. Instead, this choice should be considered as hyperparameter and optimized separately for each specific classifier.

The key contributions in this paper are summarized as follows:

- *Comprehensive Evaluation Framework:* We present a systematic testbed incorporating multiple class-balancing strategies, including both classical resampling and modern generative approaches, in conjunction with various multi-class classification models.
- *Comparative Analysis:* We compare the performance of 42 resampling combination on three datasets using various established classification metrics, ensuring a thorough analysis of the effects of different resampling methods.
- *Reproducibility:* To promote reproducibility and facilitate further research, we have made our code[3] and experimental setup publicly available, allowing for a thorough quantitative evaluation of future NIDS data augmentation approaches.

2 Related Work

Various approaches for synthetic NetFlow generation and synthetic balancing can be found in literature. Guo et al. [16] use a Conditional GAN model to generate synthetic network data for oversampling on a single dataset called ISCXVPN2016 for multi-class classification. Their goal is to use synthetic data

[3] `https://github.com/maxwolf-code/netflow_multiclass_synthetic`

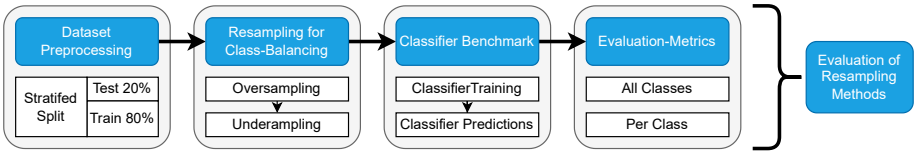

Fig. 1. The main components of the experimental setup for the comparison of resampling methods

for oversampling minority classes, thereby enhancing the performance of classification models. The setup of resampling methods consists of five traditional oversampling methods, namely Random Oversampling, SMOTE, ADASYN, SMOTE-SVM and SMOTE-TOMEK, which they compare to their CGAN approach. Only one classifier, the 1D-CNN is used as the classification model in the evaluation, measuring the classification performance via F1-Score and AUC-PR. The results show that their approach surpasses the other oversampling techniques, but their experiments are executed exclusively on a single dataset and model.

Liu et al. [26] apply a WGAN-GP model for oversampling attack NetFlows. Their approach is tested against other oversampling methods such as SMOTE, ADASYN, K-SMOTE, G-SMOTE, ACGAN-SVM, and GAN in the datasets NSL-KDD [39], UNSW-NB15 [29], and CICIDS-2017 [36]. In their experiments they test seven classifiers which are Naive Bayes, Decision Tree, Random Forest, Gradient Boosting Decision Tree, Support Vector Machine, K-Nearest Neighbors and an Artificial Neural Network. Their results suggest that the proposed oversampling method can effectively improve the detection performance. However, all the tested approaches are based on interpolation-based approaches such as SMOTE variants and ADASYN or rely on GANs, allowing for only limited comparison to other state-of-the-art generative methods. Moreover, their evaluation does not report baseline results of the classifiers when trained without resampled data. Therefore, the actual benefit of resampling remains unclear.

Liu et al. [25] apply a conditional Variational Autoencoder for oversampling. In their experiments, they use the CIC-IDS-2018 dataset exclusively and incorporate two neural network models, a 1D-CNN and a GRU Network for attack classification. Their evaluation features a comparison of the classification performance of the models that are trained on the original data and the oversampled data. They report improved macro F1 scores by around 3% and 5%. While their work highlights the effect of class balancing in the NIDS domain, their results cannot be generalized since their findings rely on a single dataset and only two classification models.

Liu et al. [22] use a diffusion-based model called NetDiffusion for the generation of packet-based network data in the pcap format. In their experiments, they use a self collected dataset consisting of video streaming, video conferencing and social media traffic. They evaluate the generated synthetic data in various aspects, e.g., by statistical and machine learning performance evaluations, network analysis or syntax related checks by parsing via wireshark or tcp-replay. In their

findings, they report that the diffusion model generates high fidelity traffic with a high conformity to protocol specifications. Moreover, they highlight that their diffusion model demonstrates potential for class balancing. While their work focused on the generation of pcap traffic, their model is based on a diffusion model, which can be adapted for NetFlow data. We therefore adapt this model and include it in our comparison.

The most similar work to ours is [17], which examines the effects of oversampling combined with undersampling, albeit exclusively on the UNSW-NB15 dataset. They report that there is no clear winning resampling combination that reliably improves classification performance. Although their work systematically tests the combination of resampling methods, they only evaluate on one dataset. Moreover, modern resampling techniques based on generative models are not tested alongside the classical approaches.

In this study, our aim is to bridge this gap by conducting a comprehensive evaluation study that compares multiple combinations of over- and undersampling including generative models with various classifiers across three datasets.

3 Experiments

In this section, we begin by providing an overview of the resampling methods under evaluation, including classical methods for oversampling, generative models, and undersampling methods. Following this, we introduce the datasets and describe our experimental setup.

3.1 Methods for Resampling

An imbalanced data distribution refers to significant differences in the number of samples across classes, which can cause classifiers to exhibit poor discriminatory power, particularly concerning minority classes [20].

Random Oversampling (ROS) is a straightforward approach to addressing imbalanced class distributions by randomly sampling from the original distributions. This involves replicating samples from the minority class(es) and adding them to the dataset to increase the number of data points in those classes [20].

Synthetic Minority Over-sampling TEchnique (SMOTE), introduced by Chawla et al. [3], is an oversampling technique that addresses imbalanced class distributions by generating synthetic data points. Unlike traditional methods that duplicate existing data, SMOTE creates new instances in the feature space using linear interpolations. For each minority class, SMOTE identifies the k nearest neighbors (kNN) from the same class. Synthetic data points are then generated along the line segments connecting these neighbors.

Adaptive synthetic oversampling (ADASYN) [21] pursues to shift the decision boundary of a classifier towards samples that are difficult to learn, as they are close to datapoints of the majority class(es). The method is based on three main steps. At first, the degree of class imbalance is calculated. Second, a kNN classifier is utilized to determine datapoints from the minority classes close

to majority classes. Next, similar to SMOTE, synthetic data points between the minority and the majority are created.

Conditional-Variational-Autoencoder (C-VAE) [37] is a generative model, based on the Variational Autoencoder which consists of an encoder network and a Decoder network coupled via a latent space, typically represented by a parameterized Gaussian distribution. The encoder $q(z \mid x)$ transforms data x into the latent space via variables z. The decoder $p(x \mid z)$ transforms the latent space z back into the data space x. The model is trained in a self-supervised manner for data reconstruction. The C-VAE extends this idea by conditioning the latent space on the output. Instead of assuming a standard Gaussian distribution for the prior of the latent representation, C-VAE uses the input observation to adjust the prior distribution of Gaussian latent variables $q(z \mid x, y)$ and the generation of the output by $p(y \mid x, z)$. Our experiments use a C-VAE trained on multi-class labels of the NetFlow data, which allows to balance the imbalanced class distribution of the original data.

Conditional-WGAN-GP (C-WGAN) [7] is based on Generative Adversarial Networks (GANs) which consist of two neural networks, a generator G and a discriminator D. Both neural networks play a min-max game, where G tries to generate realistic fake data X_f and D tries to identify given data as real or fake. Both models compete in this min-max objective, which trains the generator to generate realistic looking data from noise.

Conditional generation is performed by adding a class embedding y as input to the generator and discriminator. Therefore, G learns to model conditional data representations. In our experiments, we use a C-WGAN-GP (Conditional Wasserstein Generative Adversarial Network with Gradient Penalty) [15,1], which is trained on multi-class NetFlow data to generate class-specific samples.

Conditional Denoising Diffusion Probabilistic Model (C-DDPM) [33] incorporates a so-called diffusion process modeled as a Markov process. Diffusion-based models recently gained popularity in the image domain through high-quality image generation. At first, images are compressed in a latent dimension via an encoder model that transforms the pixel space into a lower-dimensional continuous vector representation. The latent representation can be decoded into the pixel space via a decoder. Next, noise is added to the latent representation step by step, defined by time steps t. Afterwards, a denoising U-Net neural network is trained to estimate the added noise for each time step t, which can be used to denoise the image. Eventually, the U-Net learns to denoise the latent representations, which allows to generate plausible data from randomly drawn noise input. The conditioning, originally driven by a text prompt, directs the diffusion process to generate images according to the specified conditioning. Our adapted diffusion model is trained on NetFlow data instead of images, and conditioned via class labels to generate class-specific data for oversampling.

Random Undersampling (RUS) samples data points from the minority class(es) to reduce their quantity, aiming to balance the number of data points across classes [20].

Tomek Links (Tomek) [40] are pairs of datapoints close to each other that belong to two different classes. In order to clean the decision boundaries, either both datapoints are deleted, or the datapoint of the majority classes is deleted. In our setup, we chose the removal of both data points.
Edited Nearest Neighbor (ENN) [41] is based on kNN datapoints that determine where the k-nearest datapoints of an observed datapoint belong to other classes. If so, there are two options for cleaning. One option is to exclusively delete samples of the majority classes. The second option is to delete all datapoints that are not similar to their k-nearest neighbors, which is applied in our experiments.
Neighborhood Cleaning Rule (NCR) [24] combines two main aspects, the removal of redundant data points and the cleaning of the decision boundary. The redundant data points are identified via Condensed Nearest-Neighbor Clustering, where very similar datapoints are removed. Next, based on a kNN-Classifier, ambiguous datapoints that are misclassified are removed to remove noisy data from the decision boundaries of the individual classes.
Near Miss 3 (NN-3) [28] is a heuristic cleaning procedure based on the Nearest Neighbor algorithm. NN-3 identifies the three nearest neighbors from the majority class for each example in the minority class and retains these nearest neighbors for training, thereby undersampling the majority class. By this heuristic, the algorithm keeps datapoints close to the borders of different classes that are most informative.

3.2 Benchmark datasets

The datasets applied in our experiments are originally based on packet-based datasets, which have been transformed via nProbe [30] by Sarhan et al. [35]. While other NetFlow benchmark datasets lack a common NetFlow exporter and therefore a common format with identical features, the Flow features of these datasets are homogeneous, allowing to outrule the influence of different feature sets to potential outcomes. All datasets have a heavily imbalanced class distribution in common, especially in the multi-class setting.
NF-BoT-IoT is recorded botnet traffic data of a realistic environment. Ostinato and Node-red tools were utilized to generate the non-IoT and IoT traffic. The dataset contains 586 241 (97.69%) attack NetFlows and 13 859 (2.31%) benign NetFlows. The attacks are dived into the listed types followed by the number of NetFlows: Reconnaissance 470 655, DDoS 56 844, DoS 56 833 and Theft 1 909.
NF-ToN-IoT features heterogeneous traffic based on telemetry data of Internet of Things (IoT) services, network traffic of IoT networks. The dataset includes 1 108 995 (80.4%) attack and 270 279 (19.6%) benign NetFlows. The attack types and the corresponding number of samples are: Backdoor 17 247, DoS 17 717, DDoS 326 345, Injection 468 539, MITM 1 295, Password 156 299, Ransomware 142, Scanning 21 467 and XSS 99 944
NF-UNSW-NB15 includes real network traffic and synthetic traffic based on XIA PerfectStorm. This dataset has 1 550 712 (95.54%) benign and 72 406

(4.46%) attack NetFlows. The multi-class attack types are: Fuzzers 19 463, Analysis 1 995, Backdoor 1 782, DoS 5 051, Exploits 24 736, Generic 5 570, Reconnaissance 12 291, Shellcode 1 365 and Worms 153.

3.3 Experimental Setup

Our setup comprises several multi-class classification models chosen for NetFlow classification based on current literature. To be precise, we incorporate a broad range of popular and well-known models, specifically K-Nearest Neighbors (kNN) [20], Decision Tree (DTree) [20], Random Forest (RF) [20], Extra Tree (ExTree) [11], Multi-Layer Perceptron (MLP) [20], and Extreme Gradient Boosting (XGBoost) [4].

We split each NIDS dataset into stratified train-test sets, with the training set comprising 80% of the data and the test set containing the remaining 20%. Given the extreme imbalance in the datasets, we generate stratified splits to ensure that each class is proportionally represented in both the training and test sets, maintaining the original class distributions. In accordance with the publishers of the NIDS datasets [35], we exclude IP addresses and ports, as they are strong indicators of attacks, given that certain attacks are often executed from similar IPs. Moreover, we exclude the L7_PROTO field which contains Layer 7 protocol information which are sometimes unique for normal or attack behavior and are a label indicator as well. Additionally, we normalize NetFlow attributes using min-max normalization based on the value ranges of each field of the NetFlow V9 specification [30], before applying resampling methods and classification models. The resampling methods are applied on the train set exclusively.

The objective comparison of classifier performance based on the resampling strategy requires quantitative evaluation metrics. We report Weighted-Precision, Weighted-Recall, Weighted-F1-Score, and Mattheus Correlation Coefficient [14] as they consider imbalanced class distributions to allow for an objective comparison.

4 Results

This section, describes and discusses the results of our study, first focusing on the overall classification performance, then on the individual class performance.

4.1 Evaluation of the overall classification performance

Our experiments reveal that some models show small improvements in classification performance when trained on resampled data. However, most of the results indicate lower performance compared to training on the original imbalanced training data across all tested classifier models. The heatmaps in Figure 2 display the relative improvement in classification performance for each model trained on a specific resampling combination (oversampling + undersampling), calculated by subtracting the baseline MCC values (None + None) from each

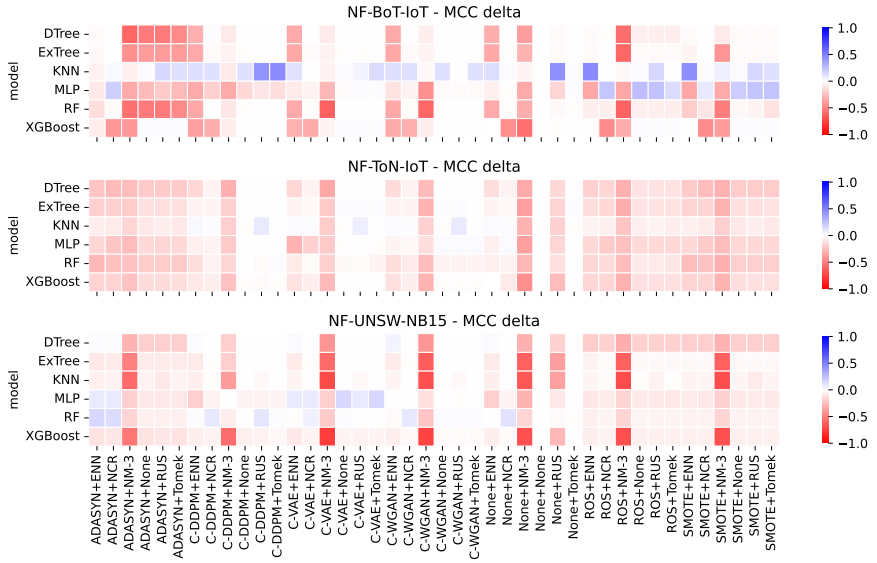

Fig. 2. Delta MCC values per model: MCC of original (None) resampling - scores of resampling strategy on the y-axis.

resampling combination score. Blue color gradients highlight improved scores, red gradients indicate decreased scores, and white indicates no difference.

The heatmap demonstrates a minimal positive impact on the classification performance. There is no consistent improvement trend where a single resampling strategy consistently improves the results of a model across all datasets. For the NF-ToN-IoT dataset a notable improvement for the kNN and MLP model can be observed. However, this can be explained by the generally poor classification performance of these models trained on the imbalanced dataset. Consequently, resampling helps these models to achieve a classification performance comparable to other classifiers. A similar pattern is seen with the MLP and RF classifiers on the NF-UNSW-NB15 dataset. Notably, some resampling methods, such as NM-3, consistently decrease performance across all datasets, suggesting that the undersampling process removes relevant information.

Next, we collected the resampling strategies with the best MCC scores for each model and dataset to analyze the total performance gain compared to the baseline reference where no resampling was applied, as shown in Table 2. The best values for each model and dataset are indicated in bold, while the best overall result for each dataset across all models is highlighted in blue. These results confirm the notable improvement mentioned above for the kNN and MLP models in the NF-Bot-IoT dataset, elevating their performance to match the MCCs of the other classification models. In general, models that already achieve high scores can be observed to exhibit minimal improvement. Although one might

infer that generative models dominate among the best results, it is important to highlight that our analysis focused exclusively on the top performing outcomes per MCC. This summary does not emphasize the general trend in which the models exhibited a decreased classification performance, as evident in Figure 2. Another major finding of Table 2 is that the XGBoost model consistently yields the best or second best results among all datasets. This finding aligns with the literature, which indicates that XGBoost achieves comparable or superior performance to state-of-the-art models across diverse domains. Our findings suggest that testing different robust classifier models is more effective for achieving good classification results compared to employing different resampling techniques.

Table 1. Listing of the numeric codes per class

Num Code	0	1	2	3	4	5	6	7	8	9
Bot-IoT	Benign	Theft	DDoS	DoS	Reco.	—	—	—	—	—
ToN-IoT	Benign	DoS	Injection	DDos	Scanning	Password	MitM	XSS	Backdoor	Ransom.
UNSW-NB15	Benign	Exploits	Reco.	DoS	Generic	Shellcode	Backdoor	Fuzzers	Worms	Analysis

4.2 Evaluation of the classification performance per class

In this section, we want to further analyze the improvement of the multi-class classification performance for each model. First, the classification performance per class using the detection rate is analyzed. The boxplots in Figure 3 show the distribution of detection rates (y-axis) per class (x-axis) among all models and applied resampling combinations. One can observe classes which can be classified reliably by many models e.g., Class 0 of NF-BoT-IoT. In contrast, classes like Class 3 and 5 of NF-ToN-IoT could not be classified correctly, and resampling did not improve the performance in these cases. Between these extremes lie classes such as Class 1, 2, and 9 of NF-ToN-IoT, which exhibit a wide range of detection rates. This suggests a strong influence of resampling for some classifiers, since e.g. Fig. 3 shows a wide range of results for the class 9 of NF-ToN-IoT for some classifier models like DTree but not for others like KNN or XGBoost.

After analyzing the overall impact on per-class performance, we proceeded to examine specific improvements across different models and datasets. Table 2 presents the detection rates per class for both the baseline and the best results. Similarly to the findings depicted in the boxplots in Figure 3, certain classes cannot be correctly classified by both the baseline models and the best models optimized by resampling. Although there are instances of improvements, for example kNN for NF-ToN-IoT in c1, where the baseline detection rate is strongly improved, these cases are exceptional. Overall, our analysis suggests that resampling may not reliably enhance classification performance for a given NIDS dataset, although some models benefit from the right choice of resampling techniques. A notable mention is XGBoost applied to the NF-UNSW-NB15 dataset,

Table 2. Comparison of baseline (None+None) vs. the best (MCC) results (in %). Bold font indicates the best of all resampling results for the model and the given dataset. Blue color indicates the best results per dataset. Detection Rate (DR in %) is given per Classes (c0 - c9), numeric class codes are explained in Table 1.

ds	model	sampling	Acc	Prec	Rec	F1	MCC	DR									
								c0	c1	c2	c3	c4	c5	c6	c7	c8	c9
NF-BoT-IoT	DTree	None+None	86.304	84.413	86.304	84.813	59.596	80	3	48	24	98	—	—	—	—	—
		C-DDPM+NCR	86.366	83.425	86.366	83.841	**60.445**	87	2	7	64	98	—	—	—	—	—
	ExTree	None+None	83.796	88.433	83.796	83.212	62.407	82	3	93	5	92	—	—	—	—	—
		ADASYN+NCR	83.629	88.030	83.629	83.312	**62.516**	91	5	93	6	91	—	—	—	—	—
	*k*NN	None+None	28.654	82.492	28.654	33.936	15.471	90	4	48	51	21	—	—	—	—	—
		ROS+None	83.456	85.731	83.456	81.943	**61.009**	91	4	91	8	21	—	—	—	—	—
	MLP	None+None	80.870	70.666	80.870	74.916	33.197	70	0	18	0	98	—	—	—	—	—
		ROS+None	80.829	87.007	80.829	**81.299**	**58.403**	84	6	5	94	88	—	—	—	—	—
	RF	None+None	86.342	84.216	86.342	83.792	60.083	79	2	8	63	98	—	—	—	—	—
		C-VAE+RUS	86.506	84.196	86.506	83.802	**60.453**	79	2	64	7	99	—	—	—	—	—
	XGBoost	None+None	83.828	87.248	83.828	85.033	60.705	84	4	27	72	92	—	—	—	—	—
		C-VAE+RUS	83.817	88.138	83.817	83.282	**62.469**	83	3	93	6	92	—	—	—	—	—
NF-ToN-IoT	DTree	None+None	49.891	43.228	49.891	**41.548**	33.991	68	0	89	19	0	0	12	0	94	3
		C-DDPM+Tomek	**49.9**	43.239	**49.9**	41.547	**34.013**	68	0	89	19	0	0	12	0	94	3
	ExTree	None+None	52.476	45.755	52.476	44.020	38.242	72	0	95	19	0	0	13	0	94	3
		None+Tomek	52.760	46.641	52.760	**44.24**	**38.834**	74	0	95	19	0	0	13	0	94	3
	*k*NN	None+None	48.053	50.256	48.053	41.219	32.908	86	1	82	3	3	8	35	0	94	96
		C-DDPM+RUS	**52.251**	50.509	**52.251**	41.583	**40.693**	87	46	95	3	0	1	35	0	94	96
	MLP	None+None	45.682	37.772	45.682	38.051	26.411	60	0	81	20	0	0	0	0	94	0
		C-WGAN+None	**46.613**	39.920	**46.613**	**39.18**	**28.188**	55	0	86	21	0	0	0	0	94	0
	RF	None+None	48.761	40.640	48.761	40.763	31.492	68	0	85	21	0	0	6	0	94	3
		C-DDPM+Tomek	**49.829**	43.184	**49.829**	**41.497**	**33.88**	68	0	89	19	0	0	6	0	94	3
	XGBoost	None+None	**56.51**	54.627	**56.51**	47.883	45.987	92	0	96	18	0	1	44	0	94	96
		C-VAE+None	56.339	53.467	56.339	47.416	**46.011**	89	0	97	18	0	0	18	0	94	96
NF-UNSW-NB15	DTree	None+None	95.812	93.753	95.812	94.356	30.901	99	19	0	4	12	0	0	3	12	0
		C-WGAN+ENN	94.509	95.338	94.509	94.312	**34.471**	98	15	50	2	8	0	16	4	6	0
	ExTree	None+None	97.009	96.612	97.009	96.502	62.027	99	85	19	16	40	0	2	26	12	0
		C-DDPM+Tomek	**97.013**	96.595	**97.013**	**96.564**	**62.508**	99	85	28	15	38	0	3	26	12	0
	*k*NN	None+None	96.992	97.289	96.992	97.074	67.749	98	78	78	26	46	43	8	50	38	6
		C-VAE+Tomek	97.011	97.272	97.011	**97.101**	**67.896**	98	79	78	26	46	44	10	50	45	7
		C-WGAN+Tomek	97.006	97.260	97.006	97.084	**67.896**	98	79	78	25	46	43	10	50	45	3
	MLP	None+None	**95.601**	92.081	**95.601**	**93.732**	19.013	99	3	1	0	0	0	0	0	0	6
		C-VAE+Tomek	93.603	95.214	93.603	93.620	**33.998**	97	0	0	0	0	0	10	62	0	64
	RF	None+None	95.638	93.938	95.638	93.751	16.626	99	8	0	1	2	0	0	3	9	0
		ADASYN+ENN	93.829	95.791	93.829	93.989	**31.552**	97	5	0	2	0	0	47	0	83	39
	XGBoost	None+None	97.406	97.225	97.406	**97.213**	**69.53**	99	84	75	18	46	13	10	32	12	0

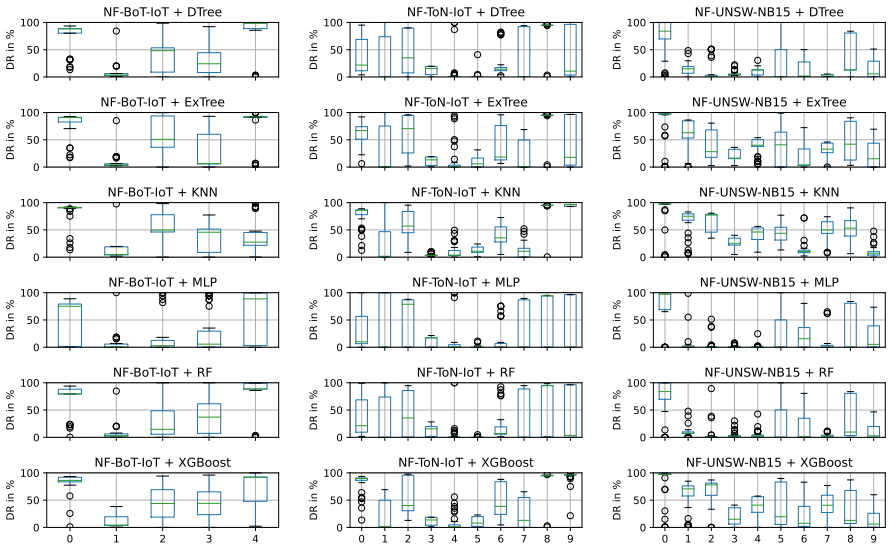

Fig. 3. Metric scores plotted as mean values of all models. (numeric class codes explained in Table 1

which performed best with the baseline data distribution across a broad spectrum of classes, although it did not achieve peak performance for each individual detection rate per class. While Fig. 3 demonstrates that some classes are challenging to classify in most cases, there are notable outliers with high performance, such as class 1 in NF-BoT-IoT or class 4 in NF-ToN-IoT. Certain resampling methods, such as NM-3, can enhance the performance for smaller, harder-to-classify classes. However, this improvement often comes at the cost of significantly reducing the detection rates for other classes, thereby lowering the overall performance, as also illustrated in Fig. 2.

5 Limitations

In this section, we acknowledge the limitations of our systematic evaluation of resampling methods. Due to the resource-intensive nature of testing multiple resampled training datesets, we selected a set of efficient and established classification models omitting some newer approaches such as E-Graph-Sage [27] in favor of XGBoost, which demonstrates comparable classification performance. Also, extensive hyperparameter evaluations per sampling combination could not be conducted due to computational limitations that might be eventually influenced by the change in class distributions. However, we believe that a fixed hyperparameter set per model allows for a fair comparison of the sampling strategies without, for example, changing the model capacity. There exist other forms of sampling bias, namely temporal bias [31] which were out of scope of our experi-

mental setup, since the tested datasets do not contain timestamps. Additionally, in the context of generative models, we had to adapt them to our datasets and therefore could not test the exact same models as they were partially trained on different data such as pcap files. Lastly, we highlight the need for thorough quality checks on generated data. Our study focused solely on assessing the impact of generated data for data augmentation. However, there are broader concerns regarding the quality of synthetic data, such as the presence of unrealistic data or the potential leakage of original data in synthetic datasets created by models that replicate training data too closely.

6 Conclusion

This work aimed to systematically analyze the influence of resampling methods for class balancing on multi-class classification models. The review of related works revealed a lack of standardization, which hinders the comparison of resampling approaches. We compiled a setup of multiple classical and state-of-the-art resampling strategies in conjunction with established multi-class classification models. Overall, we tested six classification models on 42 resampling combinations using three datasets. Our assessment of classification performance considers both overall performance and performance per class. Our results revealed a low positive impact of resampling on overall classification performance. The majority of resampling strategies had a negative impact on the classification performance across all models, suggesting that arbitrary resampling may not reliably improve multi-class classification performance. The per-class performance analysis indicated that the classification performance per class generally did not improve, as certain classes could not be classified well even after resampling. Even the resampling combinations that exhibited the greatest improvements in overall classification performance did not enhance the detection rates of certain rare classes. In conclusion, our findings suggest prioritizing the testing of different classification models, given the unreliable nature of improvements in resampling performance. In the future, our aim is to expand our study to include additional datasets and classification models. Moreover, conducting a data-driven analysis of the augmented datasets, e.g., via a NetFlow evaluation framework [42] is necessary to identify the factors contributing to both improved and diminished classification performances.

Acknowledgements This work is funded by the Bavarian Ministry of Economic Affairs, Regional Development and Energy through the GENESIS project (grant no. DIK0422/03).

References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan. arXiv (2017)
2. Boudegzdame, N., Sedki, K., Tspora, R., Lamy, J.B.: An approach for improving oversampling by filtering out unrealistic synthetic data. In: ICAART (3). pp. 291–298 (2024)

3. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research **16**, 321–357 (2002)
4. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 785–794. ACM, San Francisco California USA (2016). https://doi.org/10.1145/2939672.2939785, https://dl.acm.org/doi/10.1145/2939672.2939785
5. Elkan, C.: The foundations of cost-sensitive learning. In: International joint conference on artificial intelligence. vol. 17, pp. 973–978. Lawrence Erlbaum Associates Ltd (2001)
6. Engelmann, J., Lessmann, S.: Conditional wasserstein gan-based oversampling of tabular data for imbalanced learning. Expert Systems with Applications **174**, 114582 (2021)
7. Engelmann, J., Lessmann, S.: Conditional wasserstein gan-based oversampling of tabular data for imbalanced learning. Expert Systems with Applications **174**, 114582 (2021). https://doi.org/10.1016/j.eswa.2021.114582
8. ENISA: European union agency for cybersecurity threat landscape 2023. https://www.enisa.europa.eu/publications/enisa-threat-landscape-2023 (Oct 2023), https://www.enisa.europa.eu/publications/enisa-threat-landscape-2023
9. Fernández, A., Garcia, S., Herrera, F., Chawla, N.V.: Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. Journal of artificial intelligence research **61**, 863–905 (2018)
10. Fernández, A., López, V., Galar, M., Del Jesus, M.J., Herrera, F.: Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches **42**, 97–110 (2013). https://doi.org/10.1016/j.knosys.2013.01.018
11. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine Learning **63**(1), 3–42 (2006). https://doi.org/10.1007/s10994-006-6226-1
12. Gholampour, S.: Impact of nature of medical data on machine and deep learning for imbalanced datasets: Clinical validity of smote is questionable. Machine Learning and Knowledge Extraction **6**(2), 827–841 (2024)
13. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
14. Grandini, M., Bagli, E., Visani, G.: Metrics for multi-class classification: an overview (arXiv:2008.05756) (2020), http://arxiv.org/abs/2008.05756
15. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. Advances in neural information processing systems **30** (2017)
16. Guo, Y., Xiong, G., Li, Z., Shi, J., Cui, M., Gou, G.: Combating imbalance in network traffic classification using gan based oversampling. 2021 IFIP Networking Conference (IFIP Networking) pp. 1–9 (2021)
17. Gutiérrez, Ó.M., Núñez, J.C.S., Ávila, M., Caro, A.: A detailed study of resampling algorithms for cyberattack classification in engineering applications. PeerJ Computer Science **10**, e1975 (2024). https://doi.org/10.7717/peerj-cs.1975
18. Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., Bing, G.: Learning from class-imbalanced data: Review of methods and applications. Expert Systems with Applications **73**, 220–239 (2017). https://doi.org/10.1016/j.eswa.2016.12.035

19. Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., Bing, G.: Learning from class-imbalanced data: Review of methods and applications. Expert systems with applications **73**, 220–239 (2017)
20. Han, J., Kamber, M., Pei, J.: Data mining: concepts and techniques. Elsevier/Morgan Kaufmann, Amsterdam, 3rd ed edn. (2012)
21. He, H., Bai, Y., Garcia, E.A., Li, S.: Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence). p. 1322–1328. IEEE, Hong Kong, China (Jun 2008). https://doi.org/10.1109/IJCNN.2008.4633969, http://ieeexplore.ieee.org/document/4633969/
22. Jiang, X., Liu, S., Gember-Jacobson, A., Bhagoji, A.N., Schmitt, P., Bronzino, F., Feamster, N.: Netdiffusion: Network data augmentation through protocol-constrained traffic generation. Proceedings of the ACM on Measurement and Analysis of Computing Systems **8**(1), 1–32 (2024). https://doi.org/10.1145/3639037
23. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
24. Laurikkala, J.: Improving identification of difficult small classes by balancing class distribution. In: Quaglini, S., Barahona, P., Andreassen, S. (eds.) Artificial Intelligence in Medicine. p. 63–66. Springer Berlin Heidelberg, Berlin, Heidelberg (2001)
25. Liu, C., Antypenko, R., Sushko, I., Zakharchenko, O.: Intrusion detection system after data augmentation schemes based on the vae and cvae. IEEE Transactions on Reliability **71**(2), 1000–1010 (2022). https://doi.org/10.1109/TR.2022.3164877
26. Liu, X., Li, T., Zhang, R., Wu, D., Liu, Y., Yang, Z.: A gan and feature selection-based oversampling technique for intrusion detection. Secur. Commun. Networks **2021**, 9947059:1–9947059:15 (2021)
27. Lo, W.W., Layeghy, S., Sarhan, M., Gallagher, M., Portmann, M.: E-graphsage: A graph neural network based intrusion detection system for iot. In: NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium. pp. 1–9. IEEE (2022)
28. Mani, I., Zhang, I.: knn approach to unbalanced data distributions: a case study involving information extraction. In: Proceedings of workshop on learning from imbalanced datasets. vol. 126, p. 1–7. ICML (2003)
29. Moustafa, N., Slay, J.: Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In: 2015 Military Communications and Information Systems Conference (MilCIS). pp. 1–6 (2015). https://doi.org/10.1109/MilCIS.2015.7348942
30. Ntop: nprobe 10.1 documentation. https://www.ntop.org/guides/nprobe/index.html (2023), https://www.ntop.org/guides/nprobe/index.html
31. Pendlebury, F., Pierazzi, F., Jordaney, R., Kinder, J., Cavallaro, L.: Tesseract: Eliminating experimental bias in malware classification across space and time. In: USENIX Security Symposium (2018), https://api.semanticscholar.org/CorpusID:49905199
32. Ring, M., Schlör, D., Landes, D., Hotho, A.: Flow-based network traffic generation using generative adversarial networks. Comput. Secur. **82**, 156–172 (2018)
33. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
34. Salman, O., Elhajj, I.H., Kayssi, A., Chehab, A.: A review on machine learning–based approaches for internet traffic classification. Annals of Telecommunications **75**(11–12), 673–710 (2020). https://doi.org/10.1007/s12243-020-00770-7

35. Sarhan, M., Layeghy, S., Moustafa, N., Portmann, M.: Netflow datasets for machine learning-based network intrusion detection systems. In: Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings 10. pp. 117–135. Springer (2021)
36. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: International Conference on Information Systems Security and Privacy (2018)
37. Sohn, K., Lee, H., Yan, X.: Learning structured output representation using deep conditional generative models. Advances in neural information processing systems **28** (2015)
38. Steininger, M., Kobs, K., Davidson, P., Krause, A., Hotho, A.: Density-based weighting for imbalanced regression. Machine Learning **110**, 2187–2211 (2021)
39. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the kdd cup 99 data set. 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications pp. 1–6 (2009)
40. Tomek, I.: Two modifications of cnn. IEEE Transactions on Systems, Man, and Cybernetics **SMC-6**(11), 769–772 (Nov 1976). https://doi.org/10.1109/TSMC.1976.4309452
41. Wilson, D.L.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man, and Cybernetics **SMC-2**(3), 408–421 (1972). https://doi.org/10.1109/TSMC.1972.4309137
42. Wolf, M., Tritscher, J., Landes, D., Hotho, A., Schlör, D.: Benchmarking of synthetic network data: Reviewing challenges and approaches. Computers and Security p. 103993 (2024). https://doi.org/https://doi.org/10.1016/j.cose.2024.103993, https://www.sciencedirect.com/science/article/pii/S0167404824002980