Hindawi
Security and Communication Networks
Volume 2022, Article ID 5021125, 11 pages
https://doi.org/10.1155/2022/5021125

Research Article

Comparative Experiment on TTP Classification with Class Imbalance Using Oversampling from CTI Dataset

Heejung Kim[1] **and Hwankuk Kim**[2]

[1]*Department of Electronics Information and System Engineering, Sangmyung University, Cheonan 31066, Republic of Korea*
[2]*Department of Information Security Engineering, Sangmyung University, Cheonan 31066, Republic of Korea*

Correspondence should be addressed to Hwankuk Kim; rinyfeel@smu.ac.kr

Received 7 September 2022; Accepted 28 September 2022; Published 12 October 2022

Academic Editor: Zhe-Li Liu

Cyber threat intelligence (CTI) refers to the real-time collection of threat information and analysis of these acquired data to identify the situation and attack mechanism of a security threat. In a CTI analysis, it is important to have a standardized attack model. Recently, the MITRE adversarial tactics, techniques, and common knowledge (ATT&CK) framework has been widely used as the de facto standard security threat modeling technique. However, analyzing a large amount of data using the tactics, techniques, and procedures (TTP) of ATT&CK with a limited number of security personnel is time-consuming. To solve this cost-sensitive issue, research on automated classification of TTP from CTI data using artificial intelligence techniques is currently underway but remains challenging. This is because CTI data are domain-specific, and therefore, it is difficult to obtain labeling data to be used as training data for AI models. Hence, the distribution of training data related to TTP labeling is imbalanced. Thus, the current accuracy of ML-based TTP classification is still around 6080%. This study aims to improve the TTP classification accuracy from unstructured CTI data using machine learning while mainly focusing on solving the problems of small training datasets and TTP class imbalance. Therefore, we proposed a TTP classification method by applying easy data argumentation (EDA) and compared its performance with those of previous studies. By applying the proposed methodology, a 60–80% improvement was observed compared to the reference baseline model, TRAM. This indicates that the preprocessing methodology of applying the EDA technique is effective at improving the performance of TTP classification from unstructured CTI data in the CTI domain.

1. Introduction

The security operations center (SOC) collects security threat data to protect an organization's ICT infrastructure from internal and external cyber threats while monitoring and responding to security breaches. However, with the gradual expansion and ever-increasing number of cyberattacks, it is becoming more challenging for the SOC to promptly handle security solution events and respond to security breaches. This is because the time required to analyze a large amount of data and to provide a sophisticated response is long, and there is a dearth of skilled security personnel and resources. Security orchestration, automation, and response (SOAR) technology [1], a new paradigm of security control technology, solves these issues by automating various security threat response processes to effectively reduce repetitive tasks of security personnel and helps to quickly and accurately respond to various security events.

The core of SOAR is the integration and automation of security, orchestration, and automation (SOA), security incident response platform (SIRP), and threat intelligence platform (TIP) features [2]. SOA is a feature that interlocks and automates different workflows between numerous security solutions. The SIRP enables the automation of the process of responding to a security incident according to the response policy for each type of security incident. The TIP enables real-time collection and correlation analysis of internal and external threat data. A cyber threat intelligence (CTI) analysis is becoming crucial in quickly and effectively responding to advanced cyberattacks.

Cyber threat intelligence (CTI) data comprise of various information related to cyber threats, including information on attackers, attack procedures, and attack methods and consist of threat data analyzed by security experts, data collected from various threat sensors (such as threat data and detection data), and other related data [3]. Artificial intelligence (AI) models trained using such data are being increasingly used in the detection of new threats. In the recent past, the MITRE adversarial tactics, techniques, and common knowledge (ATT&CK) framework has often been used when analyzing cybersecurity threats and establishing a response strategy [4]. This is because the ATT&CK framework is an open-source project that is easily interoperable with other security-threat-related frameworks, such as CVE, CVSS, CAPEC, and CPE, developed by MITRE, and can be updated regularly whenever new attack techniques and patterns are discovered.

In contrast, using CTI data in conjunction with the tactics, techniques, and procedures (TTP) of MITRE ATT&CK is difficult. This is because extracting TTP information from CTI data, which are often in the form of a report, is cost-sensitive and time-consuming because CTI reports, such as the advanced persistent threat (APT) report, are unstructured threat data provided in sentence form. Manually converting these explanatory TTP sentences into the TTP naming or ID format of the ATT&CK structure is time-consuming and requires strong expertise [5]. To address these problems, there have been several efforts since 2018 to identify (extract) TTP information from CTI reports or to automatically classify the tactics and techniques in TTP.

However, several issues must be addressed to automatically increase TTP extraction or classification performance from CTI reports using AI models [6]. The first issue is insufficient training data. Training data composed of labeled TTP data, which are output data related to CTI data and are required as the input data for machine learning models, are not sufficiently available. The second issue is that of generalization error due to miss detection. As attackers constantly vary their attacks and use more advanced attack techniques, the continuous updating of TTP classification for CTI reports with new attack techniques may result in significant generalization errors and inaccurate results.

The purpose of this study is to improve TTP classification performance with insufficient training data by comparing and testing various data sampling methods.

The contributions of this study are as follows:

(i) In order to address the issues of insufficient dataset size and class imbalance in the field of CTI, two oversampling techniques, namely, synthetic minority oversampling technique (SMOTE) and easy data augmentation (EDA), were utilized, and changes in the TTP classification performance for sentence units of CTI reports were measured.

(ii) The experiment results showed better precision, recall, and $F1$ scores at the sentence level than previous works, which were reference models. An experiment with three datasets was conducted to show the generalization performance.

The structure of this study is as follows. Section 2 describes previous studies related to MITRE ATT&CK modeling and machine learning (ML)-based TTP classification. Section 3 defines the problem and describes the proposed methodology of this study. Section 4 describes the experimental design and evaluation metrics. Section 5 discusses the results and the comparative analysis with previous studies for verification of the proposed method. Finally, Section 6 describes the conclusions, implications, and future research directions.

2. Preliminary

2.1. MITRE TTP Modeling. In CTI analysis, security threat modeling is a key step for developing and evaluating defense systems against targeted attacks, such as APT attacks and spear phishing. Security threat modeling, covering the various cyber kill chains, tactics, techniques, and procedures used by attackers to carry out attacks has long been studied, and well-known examples include STRIDE, Cyber Kill Chain, and MITRE ATT&CK modeling. Table 1 shows the characteristics of the three modeling approaches.

STRIDE [7], developed by Praerit Garg and Loren Kohnfelder of Microsoft in 1999, was the first model to identify computer security threats and it was the model with the highest level of abstraction. We modeled six representative security threats that infringe on the three major elements of information protection, namely, confidentiality, integrity, and availability.

Cyber Kill Chain [8] was announced by Lockheed Martin in 2009 and is a strategic model for blocking APT attacks infiltrating the company in seven stages, namely, reconnaissance, weaponization, delivery, exploitation, installation, command and control, and exfiltration. The cyber kill chain model makes more specific attack steps than STRIDE, and defenders can utilize the cyber kill chain model when establishing a step-by-step defense strategy against APT attacks.

The MITRE ATT&CK framework [4, 9] is a modeling technique developed by MITRE in 2018. As shown in Figure 1, ATT&CK consists of tactics, techniques, and procedures related to attack techniques used to analyze the lifecycle of cyber attackers and achieve attack goals in the pre- and post-attack exploit operational stages. Currently, the enterprise ATT&CK matrix has 14 tactics and around 200 techniques (in the case of techniques, there are about 578 in total, which includes subtechniques).

2.2. Related Work. The analysis of advanced attack technologies is becoming crucial for responding quickly and effectively to intelligent cyber threats. To effectively analyze cyberattacks, the information used in cyberattacks (e.g., malicious code, IP, domain, and vulnerability), the similarity between resources, attack techniques, attack targets, and activity times should be analyzed.

To identify TTP from CTI data using ML techniques, the type of CTI data used as input data is important. CTI data can be categorized as structured data and unstructured data.

Table 1: Characteristics of the three threat modeling methodologies.

Characteristics	Stride	Cyber kill chain	MITRE ATT&CK
Source	1999, Microsoft	2009, Lockheed Martin	2018, MITRE
Level of abstraction	High	Medium	Low (detail)
Level of modeling	Attack type level	Tactics level	Tactics, techniques, and procedures
Features	Spoofing, tampering, repudiation, information disclosure, denial of service, and the elevation of privilege	Reconnaissance, weaponization, delivery, exploitation, installation, command and control, and exfiltration	Reconnaissance, resource development, initial access, execution, persistence, privilege escalation, defense evasion, credential access, discovery, lateral movement, collection, command and control, exfiltration, and impact

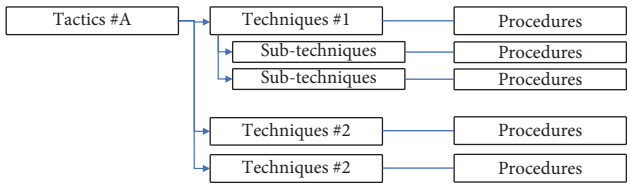


Figure 1: MITRE ATT&CK components (tactics, techniques, procedures).

Structured CTI data can express and contain TTP information in standardized formats, such as STIX, Database, and JSON, making it easier to identify TTP data from structured data than from unstructured data. However, the TTP data must be entered in advance in the specification field. Unstructured CTI data can have various forms, including reports and web pages, and when new threats arise, they are often shared in the form of reports. Therefore, studies on the use of AI and natural language processing (NLP) techniques for automated TTP identification or classification from unstructured data began in 2017.

TRAM [6] released an open-source TRAM that can automatically identify and classify TTP from CTI reports using machine learning at MITRE. This model makes the greatest contribution by disclosing proof-of-concept codes and data networks that can automatically classify tactics, techniques, and procedures with machine learning and NLP techniques. TRAM built its own dataset in which the output performs multiclassification at the techniques level of TTP by receiving input from the CTI report at a sentence unit from the input layer. The classification performance ranged between 50% and 60%.

Husari et al. proposed TTPDrill [10] and ActionMiner [11]. TTPDrill aimed to collect CTI reports from its website to identify ATT&CK techniques and CAPEC attack patterns at the document level. This approach extracts and weighs threat action-related candidate information, namely, subject, verb, and object, from each CTI report through part-of-speech tagging, and then generates 187 techniques and 19 tactics and converts them into a STIX structure. In addition, the ActionMiner model was published as a follow-up study. The purpose of this model was to find the same threat information in CTI reports by extracting object-verb pairs related to malicious software using entropy and mutual information from Wikipedia.

Legoy et al. [12] proposed the rcATT model, which is an ML model used for automatically identifying TTP from sentence units in unstructured CTI reports. This approach uses term frequency-inverse document frequency (TF-IDF) and Word2Vec as word embedding techniques, and the decision tree, support vector classifier, and AdaBoost models as classifiers. The multiclass classification performance was measured as 79.3% at the tactics level and 72.22% at the techniques level.

Ayoade et al. [13] proposed a TTP classification model using TF-IDF and support vector machines. The proposed model uses the Symantec dataset as the training dataset and APT reports as the test dataset to extract attacker actions from various CTI reports. In addition, classification performance experiments were conducted by applying various bias correction methods. The classification performance obtained was 63% at the tactics level and 96.3% at the kill chain level.

Nakanishi et al. [14] proposed the SECCMiner model. This model is not an ML-based TTP automatic classification model, and its purpose is to identify TTP-related keywords included in CTI reports using the TF-IDF NLP technique.

Kim et al. [15] proposed a technology to collect indicators of compromise (IoC) from CTI reports using NLP techniques. The IoC data and attack techniques (TTP) used for cyberattacks were extracted using the SyntaxNet technique from Google. Evaluation of 190 reports based on the $F1$ score showed an average performance of 76%.

You et al. [5] proposed a TTP intelligence mining model that extracts and classifies TTP information from unstructured CTI reports. For this model, Sentence-BERT embeddings were used in the feature extraction step, and a two-dimensional convolutional neural network and bidirectional long short-term memory network were used as classifiers, and a high $F1$ score of 0.97 was obtained. In particular, a model that focuses on embedding techniques related to the text in unstructured CTI reports was proposed. Experiments were conducted to classify six attack classes based on 6,061 TTP-related sentences for the dataset.

In summary, previous studies mainly utilized two approaches. They could be categorized as studies that aimed to find TTP and IoC data from CTI reports using various NLP techniques, and studies that classified TTP in the MITRE ATT&CK framework using ML techniques. However, the

performance of identifying threat information or classifying it as TTP showed results of 70% to 80%. In addition, previous studies have suggested that for the automation of CTI analysis, it is necessary to solve the issues relating to securing quality training data and minimizing the generalization error between training data and actual data. This indicates that research on technology to automatically identify or classify cyber threat information using AI techniques is still in the early stages and that there are many open issues to solve.

3. Proposed Model

3.1. Problem Definition. The biggest issues facing automated TTP classification in CTI are related to the quality of training data, such as small dataset size and class imbalance. The performance of ML models depends on the quality of the data for training the models. If the training data are not balanced across different classes, the performance of the ML model is significantly degraded. Although most learning models perform learning under the assumption that the proportion of the training classes is similar and provides high-performance results, however, in practice, predictive accuracy increases for classes with large data distributions and decreases for classes with small distributions, which lowers the overall performance.

CTI data are in the form of a report in unstructured text sentences. Features X, in which this type of report is entered into the TTP classification, uses sentences or documents that make up the CTI report itself as an input. Output Y can also be classified as tactics or techniques of attacking TTP. However, the biggest problem with CTI is that samples comprising input training data with TTP labels are extremely rare. This is because the CTI data and TTP information are domain-specific, and therefore, there are not much learning data for label information. Since TTP information is the result of analyzing cyber threat information by security experts, such as log information of security equipment or hacker's attack techniques, it takes several months to analyze TTP. Therefore, it takes a long time for training data with TTP labels to be opened, so it is difficult to collect training data. Moreover, it is difficult to obtain labeling datasets because ML-based TTP classification studies are in the early stages. In addition, because TTP consists of 12 tactics and 200 techniques, it has unbalanced data characteristics that inevitably result in significantly fewer data instances for each class. Currently, the only training dataset used to automatically classify TTP in unstructured CTI data is the training dataset provided by TRAM of MITRE.

3.2. Class Imbalance Issues. In supervised learning, the problem of class imbalance can arise when there is an unbalanced distribution of classes in the training dataset [16]. While imbalances in class distributions can vary, severe imbalances are more difficult to model and may require specialized skills. The general solutions for addressing unbalanced data sampling currently include oversampling and undersampling [17].

Undersampling is the process of reducing the sample size of the majority class, which has a higher proportion, to balance the amount of sample data belonging to each class. However, performance degradation may occur due to the loss of useful data because data belonging to the majority class is omitted. Oversampling is a technique that supplements the training data with multiple copies of some minority classes to increase the sample size of the minority class. Existing oversampling techniques include random oversampling, SMOTE, and data augmentation.

SMOTE [18] is an oversampling technique proposed by Chawla et al. in 2002. This technique involves the use of the k-nearest neighbor algorithm to find close neighbors of data instances belonging to the minority class and to generate virtual data through interpolation, such that the virtual data corresponds to the minority class and is not identical to the original data. In other words, a sample of a class with a small number of instances is taken and a random value is added to create a new instance, which is then added to the data [6].

EDA [19], published at the EMNLP (2019) conference, is a technique for increasing the amount of data by transforming the currently available data and is used when the amount of training data is insufficient or when a class imbalance occurs. In text classification tasks, the EDA technique improves the performance of classification models with only a small amount of data without requiring additional external data or generation models. The methods used in EDA include synonym replacement, which replaces a specific word with a synonym; random deletion, which deletes a random word; random insertion, which selects words within a sentence and inserts them into any position in the sentence; and random swap, which repositions any two words in a sentence [20, 21]. The training data can be increased in various forms using the EDA technique, which improves the performance of AI models. The above four methods can be used to produce $4 + \alpha$ augmented sentences for one sentence. It was also proved in the study that the sentences made in this way preserve the label of the original sentence, that is, the original meaning. Also, when generating sentences, noise is properly generated, which can suppress the overmatching phenomenon that may occur in data shortage problems.

Back translation [22] was first introduced in the 2016 ACL. This study attempted to improve the performance of machine translation using monolingual data. One of the methods to improve the performance of machine translation was the back translation, which was suggested to be effective. The machine translation model has an encoder-decoder structure. The source sentence is inputted to the encoder and the target sentence is inputted to the decoder and then the training of the translation model is proceeded. The author of this study proposed a methodology to create artificial source sentences with no perfect sentence format using target sentences. In other words, the original sentence is translated into another language and then retranslated to create a poor source sentence. Based on this concept, back translation techniques have been used in several studies to increase the amount of training data for performance improvement in text classification models.

In this study, the oversampling technique was used to solve the class imbalance problem of limited training data for TTP automatic classification. A small number of class distributions degrade the performance of the classification model. The reason is that the training results of the unbalanced data can bring biased results for a number of class data. However, if a small number of class data are removed and only a large number of class data are used, it may be difficult to properly classify a small number of TTP techniques. The oversampling technique was used as a methodology that can sufficiently utilize limited training data. In addition, the performance improvement results and effectiveness of the classification model using the oversampling technique were verified.

4. Experiments Design

4.1. Baseline Model. In this study, we used the TRAM model from MITRE as a baseline to compare the effectiveness of the proposed method. The TRAM model was used as the baseline model because the classifier and training dataset used were open, making it easier to compare the results in terms of the reproducibility and feasibility of the model.

4.1.1. Classifier. The classifiers used in TRAM are logic regression (LR), Naive Bayes (NB), and multilayer perception (MLP). The input X contains the CTI-related sentence text after conversion using the countVectorizer data representation method. The output Y of the classifier is the result of classifying the data into multiple classes in units of techniques of TTP.

4.1.2. Datasets. We prepared three training datasets for improving the generalization problem of the experiment. Table 2 summarizes the features of the three datasets.

The first dataset was a training dataset provided by TRAM. This training data comprised of 1,410 CTI-related sentences and 100 classes corresponding to the techniques level of TTP.

The second dataset was the data we prepared. The amount of training data provided by TRAM was small and the number of techniques to be classified was 100. Information provided by MITRE ATT&CK was collected to organize 578 techniques (including subtechniques) related to a total of 4,250 sentences. However, the number of instances per class was limited to 24. The last dataset was a combined dataset, comprising of the TRAM dataset and our dataset, which consisted of 578 techniques related to 5,660 sentences. Figure 2 shows the distribution plot of the three datasets.

As shown in Figure 2, the combined dataset data distribution was reinforced compared to the distribution of samples per TTP class provided by TRAM.

4.2. Experimental Procedure. Figure 3 shows the experimental procedure in two steps. The first step is a preprocessing stage that involves data preprocessing, sentence representation, and oversampling. Sentence representation was performed using the countVectorizer, a bag-of-words technique that expresses text as a numerical feature vector. Then, oversampling techniques such as SMOTE and EDA were applied to the training set, and the dataset was split into training and testing sets in an 8 : 2 ratio. In the second step, the classification and model evaluation was performed by training the classification model using the training set. Then, the performance of the classification model was evaluated on the testing set.

Table 2: Temperature and wildlife count in the three areas covered by the study.

	X (sentences)	Y (techniques)	X per Y
TRAM dataset	1,410	100	1–95
Proposed dataset	4,250	578	1–24
Combined dataset	5,660	578	2–103

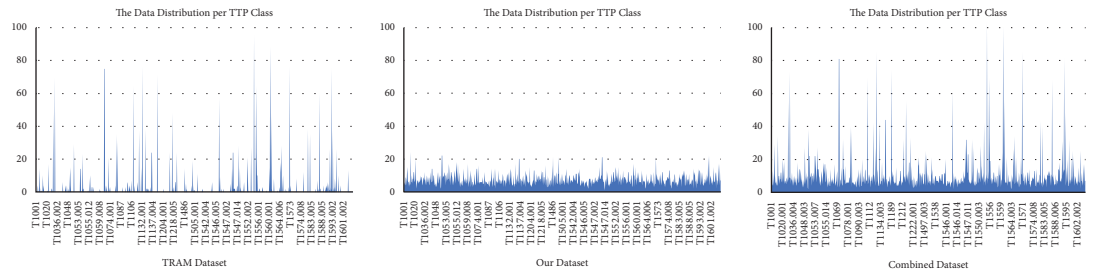
This experiment was processed in two ways. The first experiment measured the classification performance of the baseline model using the three datasets to provide a baseline for a comparative analysis. The second experiment measured the classification performance of the ML models with oversampling techniques. In this experiment, the SMOTE and EDA oversampling techniques were used.

4.3. Evaluation Methods. This section explains the evaluation method for the proposed model using our dataset. This experiment used accuracy, precision, recall, and the $F1$ score of the confusion matrix as performance indicators of the classification model. Since this experiment is a multiclassification and an imbalanced class problem, we focused on the $F1$ score and micro/macro average scores as performance metrics. In classification problems, the precision, recall, and $F1$ scores change depending on the number of instances in the target class. Therefore, we used the microaverage and macroaverage metrics, which are methods for averaging the performance for each target class and evaluating the performance of classification models with imbalanced classes. The microaverage is a method of taking the average that considers the number of instances in each class when calculating the average and is a metric that can respond sensitively to class imbalance. The macroaverage is a method of taking the average regardless of the number of instances in each class and is an indicator that can evaluate the overall performance of the model.

5. Results and Discussion

5.1. Experiment Results. This section describes the experimental results of baseline models, and the experimental results applied by oversampling techniques, SMTOE, and EDA, respectively.

5.1.1. Results with Baseline Model. This result is the baseline experiment to compare the effectiveness of our approach, oversampling techniques. The results of the first experiment are shown in Table 3. Using the training data provided by the



(a) (b) (c)

Figure 2: The distribution per TTP classes in 3 datasets: (a) dataset by TRAM, (b) our dataset, and (c) combined dataset (TRAM dataset + our dataset).

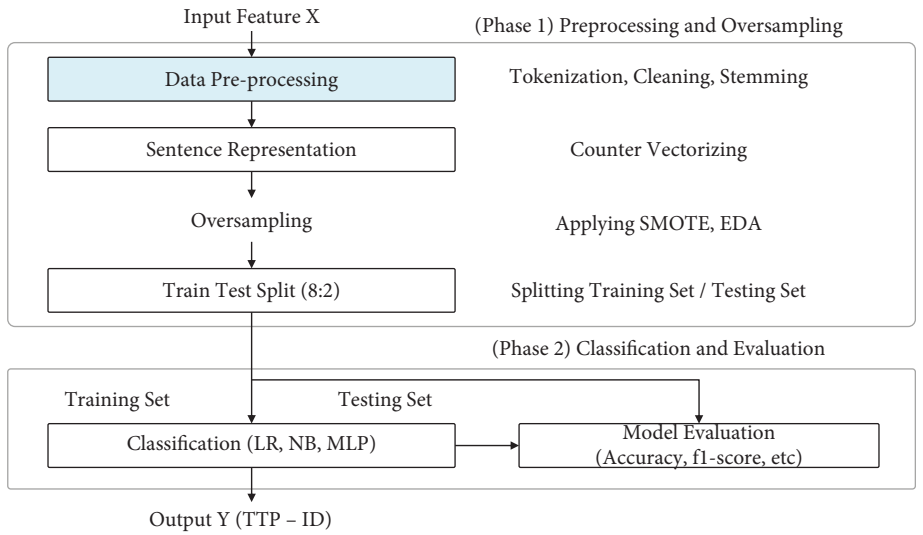


Figure 3: The procedure.

Table 3: The result of baseline experiment (%).

Data sets	Model	Accuracy (%)	Microaverage (%)			Macroaverage (%)		
			Precision	Recall	*F*1	Precision	Recall	*F*1
TRAM dataset (1,510)	LR	59.60	59.60	59.60	59.60	34.39	34.39	34.39
	NB	50.99	50.99	50.99	50.99	29.41	28.92	29.16
	MLP	61.26	61.26	61.26	61.26	39.97	38.85	39.40
	Avg.	57.28	57.28	57.28	57.28	34.59	34.05	34.32
Proposed dataset (4,250)	LR	29.06	29.06	29.06	29.06	23.09	23.55	23.32
	NB	21.29	21.29	21.29	21.29	15.09	17.06	16.02
	MLP	27.88	27.88	27.88	27.88	22.35	22.89	22.62
	Avg.	26.08	26.08	26.08	26.08	20.18	21.17	20.65
Combined dataset (5,760	LR	36.81	36.81	36.81	36.81	24.50	25.67	25.07
	NB	25.26	25.26	25.26	25.26	13.81	14.28	14.04
	MLP	35.59	35.59	35.59	35.59	26.48	28.31	27.36
	Avg.	32.55	32.55	32.55	32.55	21.60	22.75	22.16

TRAM model, which was used as the baseline model, a classification accuracy between 32.55% and 57% at the techniques level was shown. The micro-*F*1 score was between 26.08% and 57.28% and the macro-*F*1 score was between 20.65% and 34.32%. The reason for the poor performance is that the total amount of data samples is insufficient, which is affected by a small set of data, making it unsuitable.

5.1.2. Results with SMOTE. Table 4 shows the classification performance with the SMOTE sampling technique. When applying the SMOTE algorithm, the value of the neighbor *k* value parameter was set to 1. The meaning of the *K* value of 1 is the minimum number of samples per class for oversampling in SMOTE technique. Compared to the baseline model, the results of applying the SMOTE showed that the TRAM dataset was between 40.98 and 57.71%, and the

Table 4: The results of experiments with SMOTE oversampling techniques.

Datasets	Model	Accuracy (%)	Microaverage (%)			Macroaverage (%)		
			Precision	Recall	*F*1	Precision	Recall	*F*1
TRAM dataset (1,510)	LR	58.50	58.50	58.5%	58.50	43.12	44.55	43.82
	NB	56.46	56.46	56.46	56.46	38.56	40.42	39.47
	MLP	58.16	58.16	58.16	58.16	38.99	40.31	39.64
	Avg.	57.71	57.71	57.71	57.71	40.22	41.76	40.98
Proposed dataset (4,250)	LR	26.82	26.82	26.82	26.82	22.55	23.81	23.16
	NB	26.82	26.82	26.82	26.82	20.57	22.38	21.44
	MLP	25.65	25.65	25.65	25.65	20.12	22.84	21.39
	Avg.	**26.43**	26.43	26.43	26.43	21.08	23.01	22.00
Combined dataset (5,760)	LR	29.86	29.86	29.86	29.86	22.55	25.20	23.80
	NB	28.82	28.82	28.82	28.82	19.95	22.01	20.93
	MLP	26.22	26.22	26.22	26.22	19.87	22.30	21.01
	Avg.	28.30	28.30	28.30	28.30	20.79	23.17	21.92

Table 5: The result of performance with EDA oversampling techniques.

Datasets	Model	Accuracy (%)	Microaverage (%)			Macroaverage (%)		
			Precision	Recall	*F*1	Precision	Recall	*F*1
TRAM dataset (30,160)	LR	98.24	98.24	98.24	98.24	97.56	96.32	96.94
	NB	92.08	92.08	92.08	92.08	73.56	66.84	70.04
	MLP	98.76	98.76	98.76	98.76	97.27	96.83	97.05
	Avg.	96.36	96.36	96.36	96.36	89.46	86.66	88.01
Proposed dataset (84,870)	LR	93.64	93.64	93.64	93.64	93.88	93.58	93.73
	NB	85.06	85.06	85.06	85.06	88.71	82.73	85.62
	MLP	93.70	93.70	93.70	93.70	94.04	93.59	93.81
	Avg.	90.80	90.80	90.80	90.80	92.21	89.97	91.05
Combined dataset (115,030)	LR	94.75	94.75	94.75	94.75	94.31	93.70	94.01
	NB	83.61	83.61	83.61	83.61	88.58	78.29	83.12
	MLP	94.95	94.95	94.95	94.95	94.38	94.01	94.20
	Avg.	91.11	91.11	91.11	91.11	92.43	88.67	90.44

proposed dataset was between 22% and 26.43%, and the combined dataset was between 21.92% and 28.30%. The performance of the experiment with SMOTE showed little improvement compared to the reference model. Oversampling with the SMOTE technique is affected by adjacent k values, so the number of samples per class is required. Since the dataset used in this experiment contains classes with a small number of samples, it seems that the experiment was conducted with the k = 1, resulting in low performance.

5.1.3. Results with EDA. Data augmentation techniques for oversampling exist in text modification and generation methods. One of the modification methods is easy data augmentation (EDA), which augments text without external data using four text editing techniques and taking back translation and conditional pretraining as generation methods. In this study, we perform the experiment by using EDA and back translation.

Table 5 shows the classification performance with the EDA-BT (back translation) technique. Compared with the baseline model, the results of applying the EDA technique showed that the classification performance in the case of the TRAM dataset was between 88.01% and 96.36%, in the case of the proposed dataset it was between 90.80% and 91.05%, and in the case of the combined dataset it was between 90.44% and 91.11%.

5.2. Discussion: Comparative Analysis. This section compares the experimental results described in the previous section and analyses the results of previous studies and current studies.

5.2.1. Experiments Comparison. The experimental comparison of each technique is the result of comparison before and after using of oversampling with SMOTE and EDA. Here, accuracy and *F*1 score were used as predictive performance metrics and ROC-AUC metrics were used to evaluate the effectiveness of the model. As shown in Table 6, the classification results using EDA oversampling are 90% to 95% on an average, and this result shows good classification performance.

These results show that the EDA technique compared to the baseline model has significantly improved on an average in accuracy and micro/macro-*F*1 score regardless of dataset type.

Figure 4 is a graph comparing the performance results of the baseline model, applying SMOTE and EDA, respectively, with accuracy, micro-*F*1 scores, and macro-*F*1 scores for multiple classifications. Here, the *X*-axis is divided into three datasets and the *Y*-axis is the result of each performance metric. Figure 4(a) is the classification performance with the baseline model and Figure 4(b) shows

Table 6: Comparison of experiments (ACC: accuracy, macro-$F1$, and AUC: ROC-AUC).

Models		TRAM dataset			Proposed dataset			Combined dataset		
		ACC	$F1$	AUC	ACC	$F1$	AUC	ACC	$F1$	AUC
The base model	LR	59.6	59.6	0.797	29.1	29.1	0.645	36.8	36.8	0.684
	NB	51.0	51.0	0.753	21.3	21.3	0.606	25.3	25.3	0.626
	MLP	61.3	61.3	0.808	27.9	27.9	0.647	35.6	35.6	0.668
	Avg	57.3	57.3	0.786	26.1	26.1	0.633	32.6	32.6	0.659
With SMOTE	LR	58.5	58.5	0.791	26.8	26.8	0.634	29.9	29.9	0.649
	NB	56.5	56.5	0.781	26.8	26.8	0.634	28.8	28.8	0.644
	MLP	58.2	58.2	0.791	25.7	25.7	0.620	26.2	26.2	0.627
	Avg	57.7	57.7	0.788	26.4	26.4	0.629	28.3	28.3	0.640
With EDA + BT	LR	98.2	98.2	0.991	93.6	93.6	0.968	94.8	94.8	0.974
	NB	92.1	92.1	0.960	85.1	85.1	0.925	83.6	83.6	0.918
	MLP	98.8	98.8	0.994	93.7	93.7	0.969	95.0	95.0	0.975
	Avg	96.4	96.4	0.982	90.8	90.8	0.954	91.1	91.1	0.956

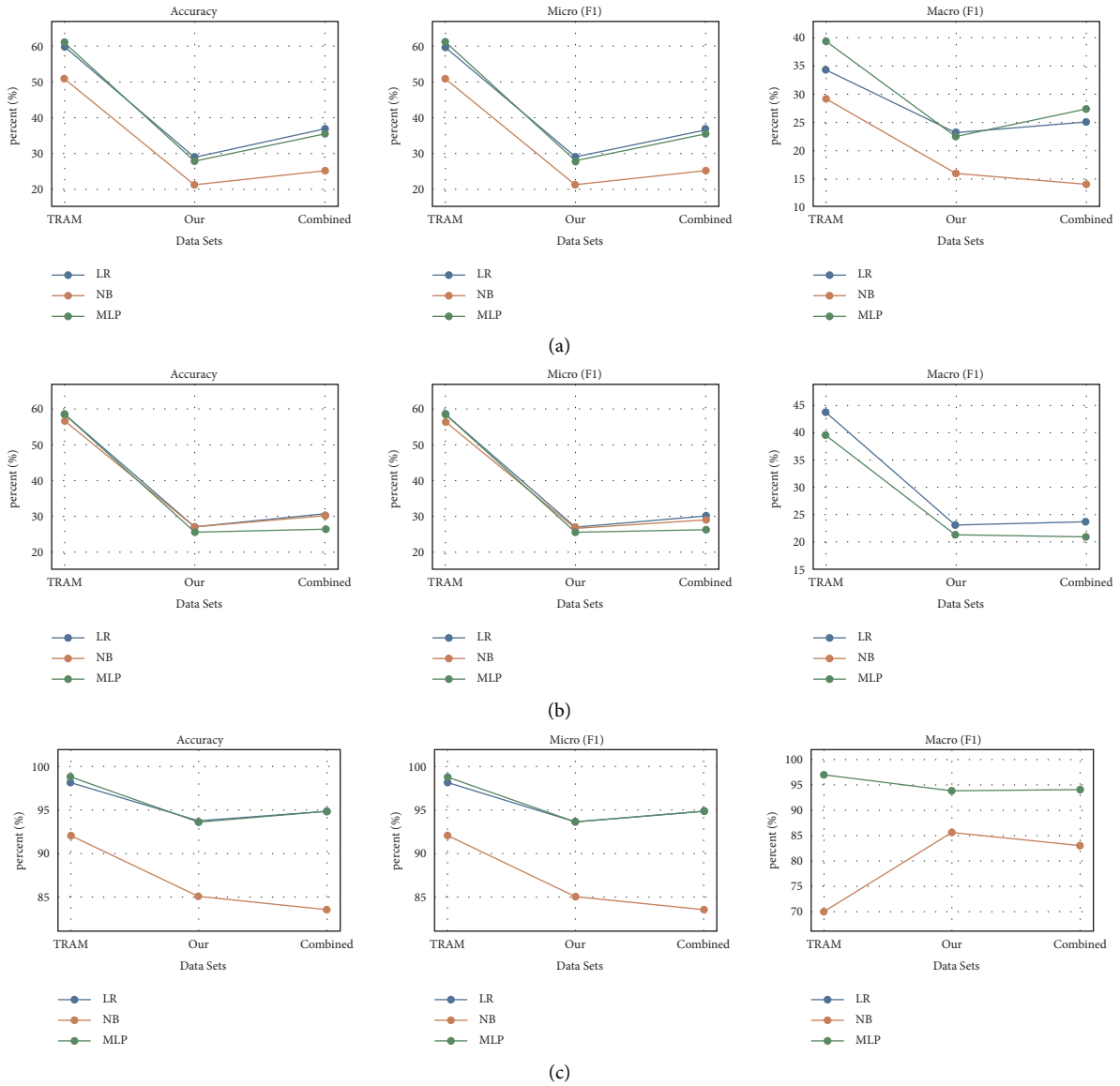


Figure 4: A graph comparing the performance results of the baseline model, applying SMOTE and EDA. (a) Performance of classification with the baseline model. (b) Performance of classification with SMOTE. (c) Performance of classification with EDA.

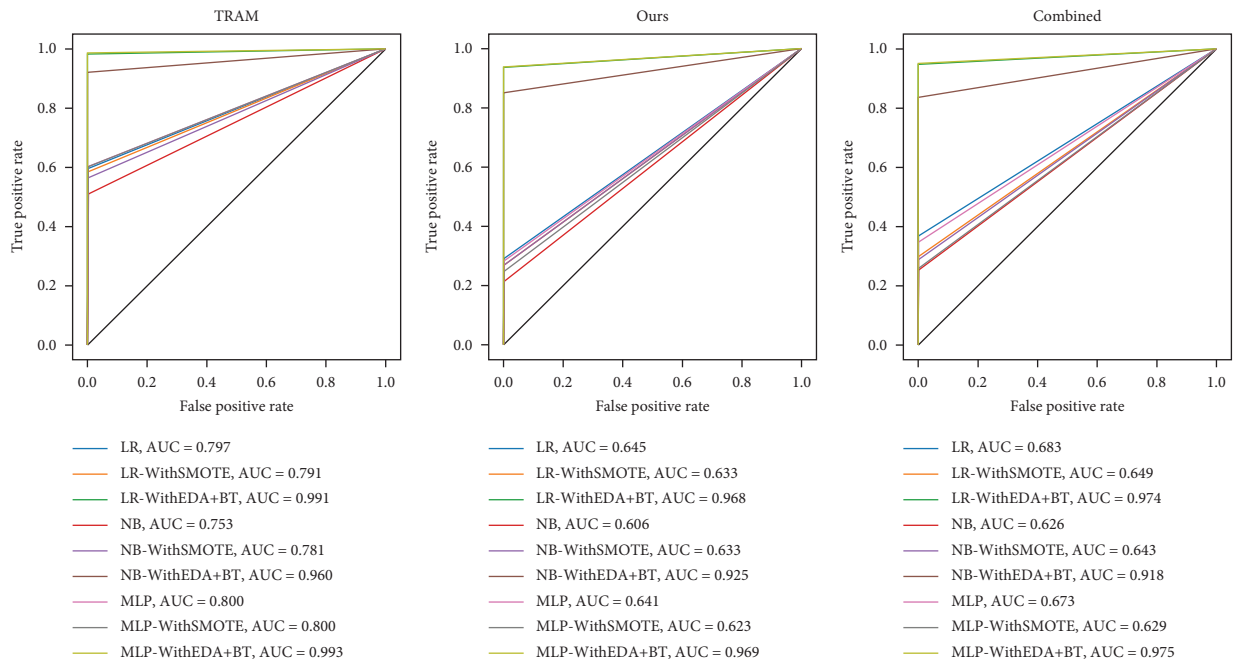


Figure 5: The result of ROC-AUC with each approach in three datasets.

Table 7: The comparison analysis with related works (legend: input = doc (document level), SEN (sentence level), metrics = ACC (accuracy), *F*1 (*F*1score), poc (proof of code), and IoC (indicator of compromise)).

Works	Objectives	Input	Approaches	Metrics	Performance
Husari et al. [10, 11]	CTI report ⟶ CAPEC ⟶ STIX conversion	DOC	NLP and ML	PoC	187 techniques 19 tactics
rcATT [12]	CTI report ⟶ TTP classification	DOC	TF-IDF, Word2Vec/AdaBoost	ACC	Tactics: 79.3% Techniques: 72.22%
Ayoade et al. [13]	CTI report ⟶ TTP attack actions	DOC	Similarity (TF-IDF)	ACC	Tactics: 63% Kill chain: 96.3%
Nakanishi et al. [14]	CTI report ⟶ TTP keywords	DOC	Similarity (TF-IDF)	PoC	445 reports
Kim et al. [15]	CTI report ⟶ IoC	DOC	NLP (SyntexNet)	*F*1	Keyword extraction: 76%
You et al. [5]	CTI report ⟶ TTP classification	SEN	Sentence-BERT embeddings/ LSTM classifier	*F*1	TTP level: 91%
TRAM (baseline)	CTI report ⟶ TTP classification	SEN	Basic ML (LR, NB, and MLP)	*F*1	Techniques: 50%~ 60%
The proposed model (EDA + BT)	CTI report ⟶ TTP classification	SEN	Oversampling/TRAM-based classifier	*F*1	Techniques: 90.8%~ 96.4%

the classification performance by using smote oversampling. Figure 4(c) shows the classification performance by applying EDA oversampling. As shown in the figure, the results of EDA improved by about 40% compared to the baseline model before applying the oversampling technique.

Figure 5 shows the ROC-AUC results of the experiment with oversampling. Comparing the ROC-AUC results, which are indicators for evaluating the discriminant power of classification models, the AUC values of SMOTE and baseline models are from 0.62 to 0.78 indicating that the discriminant performance of the model is average. In the case of EDA, the AUC value is from 0.95 to 0.98, which means that it has the best discrimination performance of the model.

5.2.2. Comparison of Previous Studies. Table 7 is the result of a comparative analysis between the current work and existing studies. As mentioned in previous studies, the comparative analysis targeted the ML-based TTP automatic classification model.

The purpose of previous studies is similar to the current study with the aim of extracting keywords of TTP from CTI reports, classifying strategies/technologies, or extracting behaviours of attacks. However, the difference from the current study is that different techniques have been applied for each study. Compared to the current approach, it is common to focus on data preprocessing techniques, but there are differences in detailed data preprocessing methods and application models, such as TF-IDF and embedding techniques.

There are also differences in input/output relationships. In five studies, including Husari et al. [10, 11], the input data were used as the document level of the CTI report. These studies aim to define the full content of the CTI report as a TTP or attack model. In three studies, including the current study, the input data were used as sentence-unit texts in the CTI report. It aims to define one sentence as a TTP or attack model. Evaluation metrics also differ from the proposed approach. Husari et al. [10, 11] and Nakanishi et al. [14] focused on implementation over evaluation metrics. In the current work, we selected *F*1 scores as evaluation metrics because we were addressing the class imbalance problem, but rcATT [12] and Ayoade et al. [13] presented evaluation metrics using accuracy indicators.

Direct comparative analysis with previous studies is difficult because the datasets, models, input/output relationships, and purposes used in each study are different. However, since the accuracy of TTP classification is not high at 20–50% on average in text-style CTI reports, and the results of previous studies to solve this problem show performance improvement at 60–90%, thus the results of this study showed good improvement compared to previous studies.

6. Conclusions

In this study, we present an automated classification of TTP from CTI data. As the occurrence of cybersecurity threats increases rapidly, it is necessary to quickly identify and respond to attacks. CTI information is mainly used to understand these threat situations and attack mechanisms. It is important to define a large amount of CTI information as a standardized attack model. However, analyzing a large amount of CTI data with a limited number of security personnel is time-consuming. Therefore, in this study, we present an automated method for classifying TTP from unstructured CTI data using machine learning. It is expected that this will enable faster identification and response to security threats.

In this study, we also focus on improving TTP classification accuracy while solving the problem of small training datasets and TTP class imbalances. Imbalanced data is one of the most important problems to be solved in machine learning. We present the comparative experimental results of TTP identification and classification performance by using data augmentation techniques during data preprocessing to address insufficient training data issues in CTI domains. As a result, when the training data augmentation technique was used based on the TRAM model, which is a reference baseline model, a performance improvement of about 60%–80% for the *F*1score was achieved.

However, a limitation of this work is that it is highly prone to generalization errors. In particular, due to the nature of the cybersecurity domain, the accuracy of ML models is bound to vary depending on the content and amount of unknown new security threats or CTI reports, as attackers continue to find new attack techniques to bypass existing defense models. Therefore, after solving this generalization error and classifying TTP from known information through rule-based matching, we believe that the proposed model can be applied to unmatched CTI information through machine learning. Future studies need to consider various improvements, such as quality training data generation, word embedding methods, model selection, and optimization, to improve automated TTP classification performance.

Data Availability

The experimental data supporting the current study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflict of interest..

Acknowledgments

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Korea Government (MSIT) (No. 2021- 0-00358, AI·Big Data Based Cyber Security Orchestration and Automated Response Technology Development).

References

[1] C. Brooks, "Security orchestration, automation and response (SOAR)-The pinnacle for cognitive cybersecurity," *Security Essentials*, p. 678, Alien Vault, 2018.

[2] F. Gartner, "Security orchestration, automation and response (SOAR)," Availablehttps://www.gartner.com/en/information-technology/glossary/security-orchestration-automation-response-soar[Online; accessed on.

[3] T. D. Wagner, K. Mahbub, E. Palomar, and A. E. Abdallah, "Cyber threat intelligence sharing: survey and research directions," *Computers & Security*, vol. 87, no. 10, pp. 101589–101592, 2019.

[4] S. Mitre, "MITRE ATT@CK framework," Available:https://attack.mitre.org/Online; accessed on, 2018.

[5] Y. You, J. Jiang, Z. Jiang et al., "TIM: threat context-enhanced TTP intelligence mining on unstructured threat data," *Cybersecurity*, vol. 5, no. 1, pp. 3–17, 2022.

[6] S. Yoder, "Automating mapping to att&ck: the threat report att&ck mapper (tram) tool," Available at:https://medium.com/mitre-attack/automating-mapping-to-attack-tram-1bb1b44bda76accessed, 2019.

[7] B. Potter, "Microsoft SDL threat modelling tool," *Network Security*, vol. 29, pp. 15–18, 2009.

[8] J. Straub, "Modeling attack, defense and threat trees and the cyber kill chain, att&ck and stride frameworks as blackboard architecture networks," in *Proceedings of the 2020 IEEE International Conference on Smart Cloud (SmartCloud)*, pp. 148–153, IEEE, Washington, DC, USA, 06-08 November 2020.

[9] A. Georgiadou, S. Mouzakitis, and D. Askounis, "Assessing mitre att&ck risk using a cyber-security culture framework," *Sensors*, vol. 21, no. 9, p. 3267, 2021.

[10] G. Husari, E. Al-Shaer, M. Ahmed, B. Chu, and X. Niu, "Ttpdrill: automatic and accurate extraction of threat actions from unstructured text of cti sources," in *Proceedings of the*

33rd annual computer security applications conference, pp. 103–115, New york, December 2017.

[11] G. Husari, X. Niu, B. Chu, and E. Al-Shaer, "Using entropy and mutual information to extract threat actions from cyber threat intelligence," in *Proceedings of the 2018 IEEE international conference on intelligence and security informatics (ISI)*, pp. 1–6, IEEE, Miami, FL, USA, 09-11 November 2018.

[12] V. Legoy, M. Caselli, C. Seifert, and A. Peter, "Automated retrieval of att&ck tactics and techniques for cyber threat reports," *Discover*, vol. 204, p. 14322, 2020.

[13] G. Ayoade, S. Chandra, L. Khan, K. Hamlen, and B. Thuraisingham, "Automated threat report classification over multi-source data," in *Proceedings of the 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pp. 236–245, IEEE, Philadelphia, PA, USA, 18-20 October 2018.

[14] A. Niakanlahiji, J. Wei, and B. T. Chu, "A natural language processing-based trend analysis of advanced persistent threat techniques," in *Proceedings of the 2018 IEEE International Conference on Big Data (Big Data)*, pp. 2995–3000, IEEE, Seattle, WA, USA, 10-13 December 2018.

[15] N. Kim, M. Kim, S. Lee et al., "Study of natural language processing for collecting cyber threat intelligence using syntaxnet," *International Symposium of Information and Internet Technology*, pp. 10–18, Springer, Cham, 2018.

[16] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Handling imbalanced datasets: a review," *GESTS international transactions on computer science and engineering*, vol. 30, no. 1, pp. 25–36, 2006.

[17] R. Mohammed, J. Rawashdeh, and M. Abdullah, "Machine learning with oversampling and undersampling techniques: overview study and experimental results," in *Proceedings of the 2020 11th international conference on information and communication systems (ICICS)*, pp. 243–248, IEEE, Irbid, Jordan, 07-09 April 2020.

[18] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 2002, pp. 321–357, 2002.

[19] J. Wei and K. Zou, "Eda: easy data augmentation techniques for boosting performance on text classification tasks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 6382–6388, EMNLP-IJCNLP), 2019.

[20] B. Li, Y. Hou, and W. Che, "Data augmentation approaches in natural language processing: a survey," *AI Open*, vol. 3, pp. 71–90, 2022.

[21] P. Liu, X. Wang, C. Xiang, and W. Meng, "A survey of text data augmentation," in *Proceedings of the 2020 International Conference on Computer Communication and Network Security (CCNS)*, pp. 191–195, IEEE, Xi'an, China, 21-23 August 2020.

[22] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," *Annual Meeting of the Association for Computational Linguistics (ACL)*, vol. 1, pp. 86–96, 2016.