Automatic Mapping of Vulnerability Information to Adversary Techniques

Otgonpurev Mendsaikhan
Graduate School of Informatics
Nagoya University
Nagoya, Japan
Email: ogo@net.itc.nagoya-u.ac.jp

Hirokazu Hasegawa
Information Security Office
Nagoya University

Yukiko Yamaguchi
and Hajime Shimada
Information Technology Center
Nagoya University

***Abstract*—Along with the growth in the usage of software in almost every aspect of human life, the risks associated with software security vulnerabilities also increase. The number of average daily published software vulnerabilities exceeds the human ability to cope with it, hence various threat models to generalize the threat landscape has been developed. The most popular threat model MITRE ATT&CK proved to be a very useful tool for the security analyst to perform cyber threat intelligence, red and blue teaming, and so on. However, for his daily operation, the security analyst has to prioritize his defense by manually mapping the daily published software security vulnerabilities to the adversarial techniques listed in MITRE ATT&CK. In this paper, we propose a method to automatically map the software security vulnerability using a multi-label classification approach. We took the vector representation of the vulnerability description and classified it with various multi-label classification methods to evaluate in different measures and found out the LabelPowerset method with Multilayer Perceptron as base classifier performs best in our experiment.**

Keywords–Multi-label classification; MITRE ATT&CK; Security Vulnerability;

I. INTRODUCTION

The digital age has presented various opportunities to society along with different challenges. One of the biggest challenges comes with the risk of cyber-attack, data breach and loss of intellectual property, and so on. Software security vulnerability is one of the biggest factors behind these challenges. According to the US National Vulnerability Database (NVD), the total number of reported vulnerability as of June 2020 is 146,000 [1] and this number is increasing year by year. In 2019 alone 20,362 vulnerabilities are reported on NVD which is a 17.6% increase from 2018 (17,308) and 44.5% increase from 2017 (14,086) and the trend is likely to be upwards [2].

Given this large number of reported vulnerabilities, tracking individual vulnerabilities is nearly impossible. Hence, there have been various approaches and threat models developed to generalize the threat landscape and to ease the burden of a security analyst. One of the most commonly used approaches is a curated knowledge base called MITRE ATT&CK® that enlists adversary behaviors including their tactics and techniques based on real-world observations. It is a powerful framework commonly used as a threat model in adversary emulation, red and blue teaming, and cyber threat intelligence practices [3]. MITRE ATT&CK generalizes the adversary attack techniques and tactics based on the common weaknesses of the systems without mentioning specific product or vulnerability.

Even though the MITRE ATT&CK proved to be a useful framework, the need to identify the specific threat that individual vulnerability poses in the adversarial landscape still exists. In layperson's terms, MITRE ATT&CK is the playbook of steps that house robber would take in order to rob a house (e.g., find open access) and software security vulnerability is the weaknesses of the house security (e.g., unlocked door or broken window). For the effective defense, the house owner needs to combine this information, the most common approaches that house robbers use and weaknesses of his house, so that he can better understand the situation and prioritize his defenses.

In this paper, we propose a method to automatically map the vulnerability information to adversary techniques and tactics. Since a specific vulnerability can be used in more than one adversarial technique we believe developing a multi-label classification model that can infer the adversarial techniques to given vulnerability would be suitable. Since every vulnerability has associated textual description, we believe using the features of this text, a classic multi-label classification algorithm could produce a result that could be useful for a practical purpose. Hence, we experimented with various multi-label classification methods to evaluate the performance to automatically map the vector representations of vulnerability description to adversary techniques and tactics as prescribed in the MITRE ATT&CK framework.

Mapping individual vulnerabilities to adversarial tactics and techniques require a certain level of expertise and domain knowledge. Thus it may consume a considerable amount of time for the security analyst. To the best of our knowledge, currently, there are no published works that directly address this problem. Therefore we believe by utilizing existing tools and data, the task of mapping vulnerability information could be automated to spare the human analyst from manual labor. Hence, the goal of the paper is to seek the possibilities to automate the mapping of vulnerability descriptions to adversarial techniques by exploring the existing tools.

The specific contributions of the paper are as follows:

1) To propose an approach to automate the mapping of vulnerability description to adversarial technique.
2) Explore and experiment with various multi-label classification methods to compare the performance.

The remainder of this paper is organized as follows. Section II will review the related research and how this paper differs in its approach. In Section III, we will briefly discuss the

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-821-1

background information to be used for this research. In Section IV, the experiment of the proposed multi-label classification and the corresponding evaluation will be discussed. Finally, we will conclude by discussing future work to extend this research in Section V.

II. Related work

There has been an attempt to use a multi-label classification approach to map cyber threat intelligence reports to adversarial techniques and tactics. Legoy et al. implemented a tool called rcATT, a system that predicts tactics and techniques related to given cyber threat reports and outputs the results using Structured Threat Information eXpression (STIX) format [4]. They focused to extract MITRE ATT&CK techniques and tactics from cyber threat reports and used simpler approaches for text representation and classification algorithms, whereas we focused to map the vulnerability description to the same framework, though using more neural and deep learning approaches.

Also, extracting general Tactics, Techniques, and Procedures (TTP) from cyber threat information is gaining some attention. Husari et al. developed a system to automate Cyber Threat Intelligence (CTI) analytics that learns attack patterns [5]. They combined Natural Language Processing (NLP) and Information Retrieval (IR) techniques to extract threat actions from threat reports based on their semantic relationships. Their focus was to extract actionable TTP from threat reports, whereas our focus is to identify the adversarial techniques that can exploit the specific vulnerability.

Apart from extracting an adversarial technique from textual documents, there have been some studies to directly map the malware behavior to the MITRE ATT&CK framework. Oosthoek et al. did the automated analysis of 951 unique families of Windows malware and mapped them onto the MITRE ATT&CK framework [6]. They generated a behavior signature of the malware in the sandbox and mapped the signature to the corresponding MITRE ATT&CK technique. Their work focused to map the malware based on its behavior to the adversarial techniques defined in MITRE ATT&CK framework whereas our focus is to map the vulnerability description that could be exploited by the adversary to the same techniques through its textual representation.

Some researchers have been working on the information provided by the MITRE ATT&CK framework to improve the adversarial predictions. Al-Shaer et al. presented their statistical machine learning analysis on Advanced Persistent Threat (APT) and software attack data reported by MITRE ATT&CK to infer and predict the techniques the adversary might use [7]. They associated adversarial techniques using hierarchical clustering with 95% confidence, providing statistically significant and explainable technique correlations. Our focus is to correlate individual vulnerability descriptions to the adversarial techniques and create a model that can be used to automatically map new vulnerability to the MITRE ATT&CK framework.

There have been also research on classifying the vulnerability information based on its textual description. Huang et al. proposed an automatic vulnerability classification model built on Term Frequency-Inverse Document Frequency (TF-IDF), Information Gain (IG), and deep neural network [8]. They validated their model with CVE descriptions of the National Vulnerability Database and compared them to the performances of SVM, Naive Bayes, and kNN algorithms. We are also attempting to classify the vulnerability information based on its textual description, but Huang et al. focused a multi-class classification that each vulnerability belongs to a specific category, whereas we attempt to classify a vulnerability into multiple adversarial techniques at the same time.

As listed above, there have been some attempts to utilize the MITRE ATT&CK framework or vulnerability classification in the academic context. However, most of the works took different directions such as [6] focused on mapping malware behavior on the adversarial technique, and [5] focused on extracting the TTPs. The only similar work [4] used some traditional methods to extract techniques from cyber threat reports, whereas we believe by using more neural and deep learning approaches, it would be possible to achieve better results.

III. Background

Since this study is on the intersection of different fields, the theoretical background knowledge is briefly explained in this section.

A. Vulnerability Modeling

There have been several attempts to standardize the reporting and modeling of software security vulnerabilities or weakness and threat landscape in general. In this section, we will discuss a few relevant schemes for this study.

1) Common Vulnerabilities and Exposures: Common Vulnerabilities and Exposures (CVE) is a list of entries, each containing an identification number, description, and at least one public reference for publicly known cybersecurity vulnerabilities [9]. CVE was launched in 1999 and now became the standard naming convention to address the interoperability and disparate databases and tools. CVE entries, also called CVEs, CVE IDs, and CVE numbers by the community provide common reference points so that cybersecurity products and services can speak the same language. CVE is an international cybersecurity community effort and each new CVE entry is assigned by CVE Numbering Authorities (CNAs).

The majority of the disclosed vulnerabilities are stored at the NVD for centralized vulnerability management purposes. The NVD is the U.S. government repository of standards-based vulnerability management data and is known as the de facto central database of software security vulnerabilities [10]. CVEs stored at NVD proved to be a useful resource for vulnerability management and overall cybersecurity-related research.

2) Common Attack Pattern Enumeration and Classification: Common Attack Pattern Enumeration and Classification (CAPEC) efforts provide a publicly available catalog of common attack patterns that helps users understand how adversaries exploit weaknesses in applications and other cyber-enabled capabilities [11]. CAPEC was established by the U.S Department of Homeland Security in 2007 and continuously evolved to include public participation and contributions. CAPEC defines “Attack Patterns” as descriptions of the common attributes and approaches employed by adversaries to exploit known weaknesses in cyber-enabled capabilities. Each attack pattern captures knowledge about how specific parts of an attack are designed and executed and gives guidance on ways to mitigate the attack’s effectiveness.

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-821-1

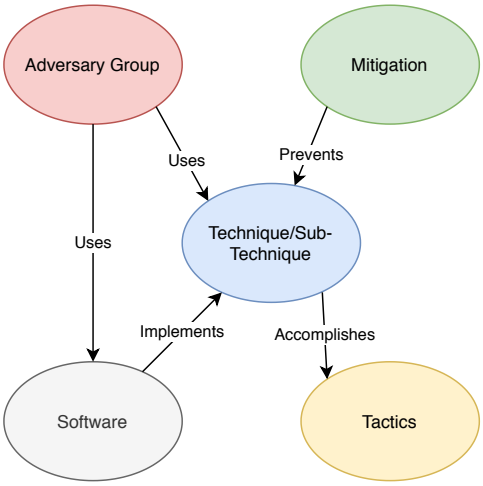

Figure 1. MITRE ATT&CK components and their relationship

CAPEC differs from MITRE ATT&CK framework in a way that it focuses on the application security and enumerates exploits against vulnerable systems, whereas the MITRE ATT&CK framework focuses on network defense and provides a contextual understanding of malicious behavior. CAPEC is mainly used for application threat modeling and developer training and education, whereas ATT&CK is used for comparing network defense capabilities and hunting new threats.

3) MITRE ATT&CK framework: Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) was created at MITRE corporation to systemically categorize adversary behavior in September 2013 [3]. It was originally designed as a project to document and categorize post-compromise adversary TTPs against Microsoft Windows systems and later added other platforms and called ATT&CK for Enterprise and publicly released in May 2015. Subsequently, complementary models such as PRE-ATT&CK, ATT&CK for Mobile, and ATT&CK for ICS has been published in 2017 and 2020. The ATT&CK framework consists of the following components:

- **Adversary group**: Known adversaries that are tracked and reported in threat intelligence reports.
- **Tactics**: Tactics represent the adversary's tactical objective: the reason for performing an action.
- **Technique/Sub-Technique**: Techniques represent "how" an adversary achieves its tactic, whereas Sub-technique further breaks down techniques into more specific descriptions of actions to reach the goal.
- **Software**: Software represents an instantiation of a technique or sub-technique at the software level.
- **Mitigation**: Mitigation represents security concepts and technologies to prevent a technique or sub-technique from being successfully executed.

The relationship between the components is visualized in Figure 1.

The MITRE ATT&CK framework is constantly enriched with techniques and sub-techniques. At the time of writing, there are 266 techniques/sub-techniques of 12 tactics in the MITRE ATT&CK Enterprise model, 174 techniques of 15 tactics in the PRE-ATT&CK model and 79 techniques of 13 tactics in ATT&CK for Mobile model.

B. Multi-label classification

Classification is the task of learning to classify the set of examples that are from a set of disjoint labels L, $|L| > 1$. If $|L| = 2$, then the learning problem is called a binary or single-label classification and if $|L| > 2$, it is a multi-class classification. In the case of multi-class classification, the example should correspond to a single class or label whereas multi-label classification the examples are associated with a set of labels $Y \subseteq L$ [12]. According to Madjarov et al. the multi-label classification methods could be of the following categories [13].

1) **Algorithm adaptation methods:** The existing machine learning algorithms that are adapted, extended, and customized for multi-label classification problem. The examples include: boosting, k-nearest neighbors, decision trees, and neural networks.
2) **Problem transformation methods:** This method transforms the multi-label classification into one or more single-label classification or regression problems. It is further divided into categories as binary relevance, label power-set, and pair-wise methods.
3) **Ensemble classification:** The ensemble methods are developed on top of existing problem transformation or algorithm adaptation methods. The examples include Random k-label sets (RAkEL) and ensembles of pruned sets (EPS) etc.

C. Evaluation measures of multi-label classification

Since the multi-label classification task is different from the traditional binary classification, the evaluation metrics to measure the performance of the method also differs. The multi-label classification measures generally fall into the following categories according to [13].

1) Example based measures
2) Label based measures
3) Ranking based measures

The evaluation measures used in this study are briefly discussed below. In the definitions, y_i denotes the set of true labels for example x_i and $h(x_i)$ denotes the set of predicted labels for the same examples. N is the number of examples and Q denotes the total number of possible class labels.

1) Subset Accuracy: Subset Accuracy, also called as Exact Match Ratio is the most strict metric, indicating the percentage of samples that have all their labels classified correctly. It can be calculated as shown in (1):

$$Accuracy(h) = \frac{1}{N}\sum_{i=1}^{N} I(h(x_i) = y_i) \quad (1)$$

where $I(true) = 1$ and $I(false) = 0$.

2) Micro averaged F1 score: Since the classification is on multiple labels the results have to be averaged out. Micro-precision and micro-recall are the measures averaged over all the example/label pair. In the definitions below TP_j, TN_j denote the number of True Positive and True Negative, FP_j,

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-821-1

FN_j denote the number of False Positive and False Negative examples per label λ_j when considered as binary classification.

$$Precision = \frac{\sum_{j=1}^{Q} TP_j}{\sum_{j=1}^{Q} TP_j + \sum_{j=1}^{Q} FP_j} \quad (2)$$

$$Recall = \frac{\sum_{j=1}^{Q} TP_j}{\sum_{j=1}^{Q} TP_j + \sum_{j=1}^{Q} FN_j} \quad (3)$$

Micro averaged F1 Score is the harmonic mean between micro-precision and micro-recall.

$$F1 = \frac{2 \times microPrecision \times microRecall}{microPrecision + microRecall} \quad (4)$$

3) Macro averaged F1 score: Macro-precision and macro-recall are the measures averaged across all labels and defined as shown in (5) and (6).

$$Precision = \frac{1}{Q} \sum_{j=1}^{Q} \frac{TP_j}{TP_j + FP_j} \quad (5)$$

$$Recall = \frac{1}{Q} \sum_{j=1}^{Q} \frac{TP_j}{TP_j + FN_j} \quad (6)$$

Macro-F1 is the harmonic mean between precision and recall where the average is calculated per label and then averaged across all labels. If P_j and R_j are the precision and recall for all $\lambda_j \in h(x_i)$ from $\lambda_j \in y_i$ then Macro F1 is defined as in (7):

$$F1 = \frac{1}{Q} \sum_{j=1}^{Q} \frac{2 \times P_j \times R_j}{P_j + R_j} \quad (7)$$

4) Hamming loss: Hamming loss evaluates how many times an example-label pair is misclassified i.e., fraction of labels that are incorrectly predicted.

$$HammingLoss(h) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{Q} \mid h(x_i) \Delta y_i \mid \quad (8)$$

where Δ stands for the symmetric difference between two sets. The smaller the Hamming loss better the model performance.

5) Ranking loss: Ranking loss evaluates the average fraction of label pairs that are reversely ordered for the particular example.

$$RankingLoss(h) = \frac{1}{N} \sum_{i=1}^{N} \frac{\mid D_i \mid}{\mid y_i \mid \mid \bar{y_i} \mid} \quad (9)$$

where

$D_i = \{(\lambda_m, \lambda_n) \mid f(x_i, \lambda_m) \leq f(x_i, \lambda_n), (\lambda_m, \lambda_n) \in y_i \times \bar{y_i})\}$, while $\bar{y_i})\}$ denotes the complementary set of y in L. The smaller the Ranking loss better the model performance.

IV. Experiment

Using the background information of Section III we conducted the experiment on the multi-label classification of vulnerability information.

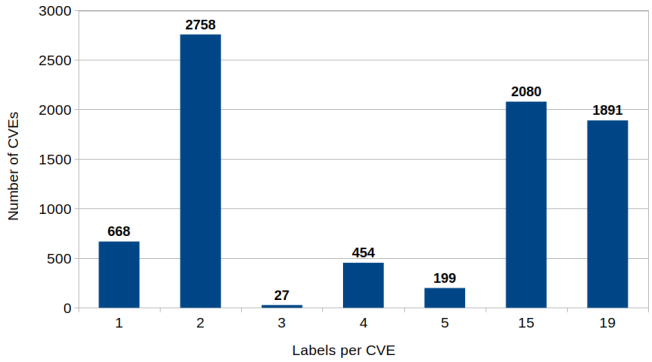

Figure 2. CVE to Label mapping of the dataset

A. Experimental Dataset

The European Union Agency for Cybersecurity (ENISA) published a report in December 2019 titled State of Vulnerabilities 2018/2019 [14]. The report aimed to provide an insight into both the opportunities and limitations of the vulnerability ecosystem. They collected in total 27,471 vulnerability information published during 1st January 2018 to 30th September 2019 from various data sources. As part of the analysis of the collected data, authors mapped the CVEs to the MITRE ATT&CK technique using the common CAPEC information found in both NVD and ATT&CK. The authors generously made available the dataset they've analyzed [15] and we utilized the CVE information mapped to MITRE ATT&CK tactics and techniques for training and testing the multi-label classification model.

The ENISA report dataset contained 8,077 CVEs that are mapped to 52 unique MITRE ATT&CK techniques or in this instance labels. The dataset cardinality (mean of the number of labels of the instances) is 9.43 and density (mean of the number of labels of the instances that belong to the dataset divided by the number of dataset labels) is 0.18. The dataset CVEs are distributed into 7 discrete buckets of technique combinations. For example, there are 668 CVEs that have a single label or technique associated with it and 1,891 CVEs that have 19 labels assigned to them. Figure 2 shows this distribution.

From the ENISA report dataset of 8,077 examples, we held out 200 examples to validate and analyze the trained model. The remaining 7,877 examples were used to train and evaluate the various multi-label classification methods.

In this study, we focused to do multi-label classification based on the textual features of the CVE descriptions. The mean length of the CVE description is 368 characters and minimum/maximum lengths are 40 and 3,655 characters long. The oldest CVE updated during the data collection period is CVE-2007-6763 and the newest is CVE-2019-9975.

B. Text representation

In order to conduct the multi-label classification, the given text needs to be converted into numerical vectors, also known as embeddings. Conventionally, vector embeddings were achieved through shallow algorithms such as Bag of Words (BoW) or Term Frequency-Inverse Document Frequency (TF-IDF). These approaches have been superseded by predictive representation models such as Word2Vec [16],

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-821-1

GloVe [17], and so on. Since the utilization of deep neural networks has been proven to be superior in different fields, various studies have adopted deep neural models to embed the text into vector space, such as Facebook's InferSent [18] and Universal Sentence Encoder (USE) from Google Research. Perone et al. evaluated different sentence embeddings and Universal Sentence Encoder outperformed InferSent in terms of semantic relatedness and textual similarity tasks [19]. Therefore, for the purpose of this research, Universal Sentence Encoder has been utilized to generate the vector embeddings of the text.

Since the sentence embeddings from USE produce good task performance with little task-specific training data, a Deep Averaging Network (DAN)-based USE model introduced in [20] has been used to represent the CVE descriptions in numerical vectors, so that multi-label classification methods could be applied. The model takes English sentences of variable lengths as input and produces 512 fixed-dimensional vector representations of the sentences as output [21].

C. Model selection

The CVE descriptions of the dataset have been initially converted into numerical vectors using USE. Every CVE description becomes a 512 fixed-dimensional vector that can be treated as features for the classifier. To determine the suitable model to map the vulnerability description to MITRE ATT&CK techniques we experimented with 1 Algorithm Adaptation, 3 Problem Transformation, and 1 Ensemble multi-label classification methods using the open-source library scikit-multilearn [22]. The experimented methods are listed below.

- **Multi-label k-nearest neighbors (MlkNN)** is the adaptation of the popular k-nearest neighbors (kNN) algorithm to the multi-label classification task and an example of Algorithm adaptation method. We estimated the number of neighbors k to be most optimal when $k = 3$ where $1 \leq k \leq 30$ when optimized for macro-average F1 measure.
- **LabelPowerset** is a Problem Transformation method that transforms a multi-label problem to a multi-class problem with 1 multi-class classifier trained on all unique label combinations. It maps each combination to a unique id number and performs multi-class classification using the classifier as a multi-class classifier and combination ids as classes.
- **ClassifierChain** is also a Problem Transformation method. The classifiers are linked along a chain where the i-th classifier deals with the binary relevance problem associated with its label. The feature space of each link in the chain is extended with the 0/1 label associations of all previous links.
- **BinaryRelevance** is the well known one-against-all method. It learns one classifier for each label using all the examples labeled with that label as positive and remaining as negative. And while making a prediction each binary classifier predicts whether its label is relevant for the given example or not. It is an example of the Problem Transformation method.
- **RAndom k-labELsets multi-label classifier (RAkELd)** is an Ensemble method that divides the label space into equal partitions of size k, trains a LabelPowerset classifier per partition and predicts by summing the result of all trained classifiers.

The above-mentioned methods use traditional classification algorithms for the multi-label classification task. Since the utilization of neural networks has been proven to be superior in almost every task we also experimented with more neural approaches as multi-label classification algorithms. Since the multi-label classification task of our experiment doesn't require the sequential input or memory state of the input we experimented with a simple Multilayer Perceptron (MLP) neural model to conduct the classification. Szymański et al. included a wrapper in a scikit-multilearn library that allows any Keras or PyTorch compatible backend to be used to solve multi-label problems through problem-transformation methods [23]. We utilized it to conduct the same experiment with neural methods. The following lists the neural methods we used along with their basic parameters.

- **LabelPowerset (neural)** LabelPowerset method with the Multilayer Perceptron as the base classifier. It has 2 hidden layers and the softmax function is used for activation.
- **BinaryRelevance (neural)** BinaryRelevance method with the Multilayer Perceptron as base classifier. It has 2 hidden layers and the sigmoid function is used for activation.

In the next section, we will discuss the evaluation results of these different methods.

D. Model Evaluation

Since the dataset is limited in size we evaluated the models with 10-fold cross validation method. The evaluations results as the average of 10-fold validation is listed in Table 1.

From the results listed in Table 1, we could see that LabelPowerset using the neural model as base classifier has the best results in all except one evaluation measures we selected. In terms of Hamming loss, the neural BinaryRelevance has a better score, but neural LabelPowerset is second in place and outperforms in every other measure. Hence, neural LabelPowerset model has been chosen as the best performing model.

There is no rule-of-thumb for "good" multi-label classification result and it depends upon various factors such as classification domain, dataset, and evaluation measures. However, in order to understand the efficiency of the model we compared our results with the results published in [24] in which Pakrashi et al. did a benchmarking study on various multi-label classification algorithms using eleven different datasets. When compared with their results it has revealed that the least performance in our experiment is better than the best performing results of 4 of the 11 datasets in the same measure as Macro Averaged F1 score. Hence, we concluded that the experimental results are good enough to be considered for discussion.

E. Model analysis

After training and testing the best performing model we conducted an in-depth analysis of the model using the held-out validation data. A total of 200 examples were held out from the training dataset to validate and analyze the best performing

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-821-1

TABLE I. EXPERIMENT RESULT.

		Micro Average			Macro Average				
Algorithm	*Accuracy score*	*Precision*	*Recall*	*F1 Score*	*Precision*	*Recall*	*F1 Score*	*Hamming loss*	*Ranking loss*
MlkNN	0.6138	0.7376	0.6211	0.6740	0.6507	0.5081	0.5576	0.1079	0.3595
LabelPowerset	0.6133	0.7186	0.5741	0.6369	0.5753	0.5412	0.5174	0.1157	0.3654
ClassifierChain	0.5036	0.5978	0.6208	0.6089	0.4209	0.4715	0.4243	0.1427	0.4298
BinaryRelevance	0.3744	0.5798	0.6576	0.6158	0.4638	0.6193	0.4907	0.1471	0.3263
RakelD	0.4237	0.6255	0.6216	0.6230	0.5021	0.5884	0.5024	0.1340	0.3411
LabelPowerset (neural)	**0.7432**	0.7532	0.7380	**0.7452**	0.6827	0.6264	**0.6396**	0.0911	**0.2448**
BinaryRelevance (neural)	0.5538	0.7789	0.7100	0.7426	0.6924	0.5957	0.6279	**0.0883**	0.2885

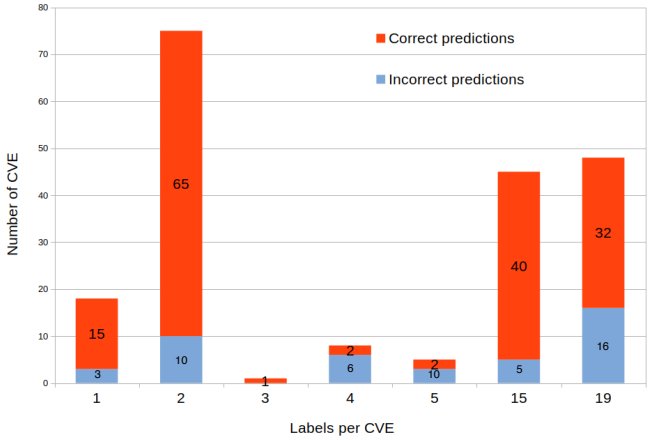

Figure 3. CVE to Label mapping of validation dataset of 200 examples

model. Neural LabelPowerset model trained and tested with 7,877 examples is used to predict the labels for the previously unseen 200 examples. Figure 3 depicts the prediction result in terms of distributions of CVEs per label. In this experiment, we considered the correct prediction only if all the expected labels are correctly predicted.

From Figure 3, it could be seen that when the number of labels to be predicted increases, the chances of incorrect prediction also increases. We believe such bias could be as a result of the skewed training data. Since the experiment dataset is small in size and limited to only 52 adversarial techniques of 266 techniques/sub-techniques of MITRE ATT&CK Enterprise model and distributed to only 7 different buckets of the number of labels (see Figure 2), the model tends to inadequately classify the examples. However, based on this analysis, we believe with a comprehensive training dataset, the multi-label classification method could be applied to the mapping of vulnerabilities to the adversarial techniques.

V. Conclusion

In this paper, we proposed an approach to automatically map the vulnerability information to adversary techniques in the cybersecurity context. We converted vulnerability descriptions into vector space and experimented with various multi-label classification methods to identify the most suitable method to map the vulnerability into MITRE ATT&CK adversarial techniques. We used 8,077 examples from open datasets prepared by ENISA, of which 7,877 have been used to train and test 7 multi-label classification methods in 9 evaluation measures. We also did a comprehensive analysis of the remaining 200 examples as a prediction only task using the best performing neural LabelPowerset model.

Due to the partial nature of the experimental dataset, the experimental result could not be fully tested in real-life scenarios. However, in the given dataset, the chosen methods show good performance, indicating a comprehensive dataset may yield a production-ready system that could be used to automate and prioritize the cyber defense operations.

In the future, we would like to build a comprehensive dataset by correlating CAPEC information of the vulnerability with MITRE ATT&CK techniques and create a model that can be used in production systems.

References

[1] National Vulnerability Database, 2020 (accessed June 1, 2020). [Online]. Available: https://nvd.nist.gov/general/nvd-dashboard

[2] D. Bekerman and S. Yerushalmi, The State of Vulnerabilities in 2019, 2020 (accessed June 10, 2020). [Online]. Available: https://www.imperva.com/blog/the-state-of-vulnerabilities-in-2019/

[3] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "MITRE ATT&CK®: Design and philosophy," The MITRE Corporation, Tech Report, 2020.

[4] V. Legoy, M. Caselli, C. Seifert, and A. Peter, "Automated retrieval of att&ck tactics and techniques for cyber threat reports," ArXiv, vol. abs/2004.14322, 2020.

[5] 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, ser. ACSAC 2017. New York, NY, USA: Association for Computing Machinery, 2017, p. 103–115. [Online]. Available: https://doi.org/10.1145/3134600.3134646

[6] K. Oosthoek and C. Doerr, SoK: ATT&CK Techniques and Trends in Windows Malware, 12 2019, pp. 406–425.

[7] R. Al-Shaer, J. M. Spring, and E. Christou, "Learning the associations of mitre att&ck adversarial techniques," 2020.

[8] G. Huang, Y. Li, Q. Wang, J. Ren, Y. Cheng, and X. Zhao, "Automatic classification method for software vulnerability based on deep neural network," IEEE Access, vol. 7, 2019, pp. 28 291–28 298.

[9] Common Vulnerabilities and Exposures, 2020 (accessed June 25, 2020). [Online]. Available: https://cve.mitre.org/

[10] National Vulnerability Database, 2020 (accessed June 22, 2020). [Online]. Available: http://nvd.nist.org

[11] About CAPEC, 2020 (accessed June 25, 2020). [Online]. Available: https://capec.mitre.org/about/index.html

[12] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," IJDWM, vol. 3, 2007, pp. 1–13.

[13] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Deroski, "An extensive experimental comparison of methods for multi-label learning," Pattern Recogn., vol. 45, no. 9, Sep. 2012, p. 3084–3104. [Online]. Available: https://doi.org/10.1016/j.patcog.2012.03.004

[14] V. Katos, S. Rostami, P. Bellonias, N. Davies, A. Kleszcz, S. Faily, A. Spyros, A. Papanikolaou, C. Ilioudis, and K. Rantos, "State of vulnerabilities 2018/2019," European Union Agency for Cybersecurity (ENISA), Tech Report, 2019.

[15] ENISA's state of vulnerabilities 2018/2019 report, 2019 (accessed May 10, 2020). [Online]. Available: https://github.com/enisaeu/vuln-report/

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-821-1

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in 1st International Conference on Learning Representations, 2013. [Online]. Available: http://arxiv.org/abs/1301.3781

[17] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[18] InferSent, 2018 (accessed January 10, 2020). [Online]. Available: https://github.com/facebookresearch/InferSent

[19] C. S. Perone, R. Silveira, and T. S. Paula, “Evaluation of sentence embeddings in downstream and linguistic probing tasks,” CoRR, vol. abs/1806.06259, 2018. [Online]. Available: http://arxiv.org/abs/1806.06259

[20] D. Cer, Y. Yang, S. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, Y. Sung, B. Strope, and R. Kurzweil, “Universal sentence encoder,” CoRR, vol. abs/1803.11175, 2018. [Online]. Available: http://arxiv.org/abs/1803.11175

[21] universal-sentence-encoder, 2018 (accessed January 11, 2020). [Online]. Available: https://tfhub.dev/google/universal-sentence-encoder/2

[22] Multi-Label Classification in Python, 2018 (accessed May 15, 2020). [Online]. Available: http://scikit.ml/

[23] P. Szymański and T. Kajdanowicz, “A scikit-based Python environment for performing multi-label classification,” ArXiv e-prints, Feb. 2017.

[24] A. Pakrashi, D. Greene, and B. MacNamee, “Benchmarking multi-label classification algorithms,” in 24th Irish Conference on Artificial Intelligence and Cognitive Science (AICS’16), Dublin, Ireland, 20-21 September 2016. CEUR Workshop Proceedings, 2016.

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-821-1