ST-152 Workshop on **Intelligent Autonomous Agents for Cyber Defence and Resilience**

Adversarial Deep Learning Against Intrusion Detection Classifiers

Maria Rigaki*, Ahmed Elragal**, Luleå University of Technology

**Sigurd Hoels vei 100, 0655 Oslo, Norway, +47 40551802, marrig-9@student.ltu.se*

**Campus Luleå, A3412 Luleå, Sweden, +46 (0)920 493670, ahmed.elragal@ltu.se

Abstract

Traditional approaches in network intrusion detection follow a signature-based approach, however the use of anomaly detection approaches and machine learning techniques have been studied heavily for the past twenty years. The continuous change in the way attacks are appearing, the volume of attacks, as well as the improvements in the big data analytics space, make machine learning approaches more alluring than ever. The intention of this paper is to show that using machine learning in the intrusion detection domain should be accompanied with an evaluation of its robustness against adversaries. Several adversarial techniques have emerged lately from the deep learning research, largely in the area of image classification. These techniques are based on the idea of introducing small changes in the original input data in order to make a machine learning model to misclassify it. This paper follows a big data analytics methodology and explores adversarial machine learning techniques that have emerged from the deep learning domain, against machine learning classifiers used for network intrusion detection. We look at several well-known classifiers and study their performance under attack over several metrics, such as accuracy, F1-score and receiver operating characteristic. The approach used assumes no knowledge of the original classifier and examines both general and targeted misclassification. The results show that using relatively simple methods for generating adversarial samples it is possible to lower the detection accuracy of intrusion detection classifiers as much as 27%. Performance degradation is achieved using a methodology that is simpler than previous approaches and it requires only 6.14% change between the original and the adversarial sample, making it a candidate for a practical adversarial approach.

Keywords: Adversarial machine learning, intrusion detection, big data analytics

CEUR-WS.org/Vol-2057/Paper7.pdf

1 Introduction

Despite the security measures deployed in enterprise networks, security breaches are still a source of major concern. Intrusion detection is dealing with unwanted access to systems and information by any type of user or software. There are two major categories of IDS: Network IDS (NIDS), which monitor network segments and analyze network traffic at different layers in order to detect intruders and Host based IDS (HIDS), which are installed in host machines and they try to determine malicious activities based on different indicators such as processes, log files, unexpected changes in the host and so on. The focus in this paper is on NIDS.

In large enterprise networks the amount of network traffic that is generated on a daily basis, requires consideration about gathering, storing and processing of said traffic. While one approach is to discard parts of the data or log less information, the emergence of Big Data Analytics (BDA) as well as the improvement in computing power, memory and the decrease in storage costs, transforms the situation into a big data problem.

Traditional approaches in the area of intrusion detection mainly revolve around signature / misuse approaches which have the limitation that they work only with known attack patterns and that they require extensive domain knowledge. Anomaly detection techniques based on statistical or machine learning approaches promise more flexibility and less dependency in domain knowledge and are more scalable when it comes to big data. Using BDA methods seems like a very likely approach as the amount and speed of data generated is expected to increase further in the future. However, one has to question not only the performance of the BDA methods proposed, but also their stability and robustness against adversaries that will most certainly try to attack them.

In this paper, we utilize adversarial machine learning methods against machine learning classifiers that are used for NIDS. The methods were previously introduced in image based datasets and we examine their suitability in the NIDS domain in terms of ability to degrade machine learning classifier performance and in terms of practical value. We also present some initial results regarding classifier robustness against the most promising methods and under the specified threat model, as well as a potential feature selection method that can be used by attackers in order to masquerade their traffic as normal.

2 Related Work

Despite the numerous research activities around machine learning and intrusion detection, there has been substantially less focus on adversarial machine learning, i.e. the robustness of the machine learning methods in the face of adversaries. A qualitative taxonomy for the threat models against machine learning systems was introduced by Barreno et al. (2006). It placed the attacks in three axes: *Influence* (causative or exploratory), *security violations* (integrity, confidentiality) and *specificity* (indiscriminate or targeted). The same taxonomy was used by Huang et al. (2011) and was extended further to include privacy as a security violation when the adversary is able to extract information from the classifier.

Another taxonomy was introduced by Papernot et al. (2016a) and focuses on two axes: *Complexity* which ranges from simple confidence reduction to complete source / target misclassification and *knowledge* which ranges from knowledge about architecture, training tools and data to just knowledge of a few samples. If the attacker knows anything regarding the architecture, the training data or the features used, the attack is considered a white-box attack. If the adversary's knowledge is limited to Oracle attacks or she has only access to limited number of samples, the attack is considered a black-box attack.

Viewing the problem from the attacker perspective, attacks can also be categorized as poisoning or evasion ones. Different poisoning attacks have been described in Biggio et al. (2012) and Xiao et al. (2015). Both studies try to poison the training data in different ways. Xiao et al. (2015) devised attacks against linear classifiers such as Lasso and Ridge by maximizing the classification error with regards to the training points, while Biggio et al. (2012) attacked Support Vector Machines (SVM) by injecting samples to the training set in order to find the attack point that will maximize the classification error.

Evasion attacks have been studied by Ateniese et al., (2015), Biggio et al. (2010) and Biggio et al (2013). The latter proposed a methodology which requires the generation of multiple training sets and subsequently the creation of several classifiers which are combined to create a meta-classifier. This meta-classifier is used in order to extract statistical properties from the data but not the features themselves, which makes it an attack against privacy.

Deep Learning (DL) methods have been wildly successful in recent years especially in areas such as computer vision and speech recognition. As part of this development, Adversarial Deep Learning (ADL) have also surfaced, mostly centered around the computer vision domain. Szegedy et al. (2013) showed that making very small variations in an image, one could fool a Deep Learning model to misclassify it. The variations can be small enough that can be imperceptible to humans.

Several methods of producing adversarial samples have been proposed so far which trade on complexity, speed of production and performance:

- Evolutionary algorithms were proposed by Nguyen et al. (2015) but the method is relatively slow compared to the two other alternatives.
- Fast Gradient Sign Method (FGSM) proposed by Goodfellow et al., (2014).
- Jacobian-based Saliency Map Attack (JSMA) (Papernot et al., 2016a) is more computationally expensive than the fast gradient sign method but it has the ability to create adversarial samples with less degree of distortion.

It is not only Deep Learning models that are vulnerable to adversarial samples produced by the above methods. Shallow linear models are also plagued by the same problem and so are model ensembles. The only models that have shown some resistance to adversarial samples are Radial Basis Function (RBF) networks, however, they cannot generalize very well (Goodfellow et al., 2014). The concept of transferability was thoroughly tested by Papernot et al. (2016b). The

authors tested several classifiers both as source for adversarial sample generation as well as target models. However, the testing was confined to image classifiers.

When it comes to ADL, the domain of the different attacks and adversarial sample generation has mainly revolved around the area of image classification and computer vision. Recent work has shown that it is also possible to create adversarial samples against neural network malware classifiers (Grosse et al., 2016). Other applications of general AML in security involve spam classifiers (Nelson et al., 2008; Zhou et al., 2012; Huang et al., 2011), malware analysis (Biggio et al., 2014; Grosse et al., 2016), biometrics (Biggio et al., 2010) and network traffic identification (Ateniese et al., 2015). There are also two studies related to Intrusion detection (Biggio et al., 2010; Huang et al., 2011) and they both assume a causative influence model.

This paper follows a different approach than most of the previous work in the adversarial machine learning field which was either addressing poisoning attacks or was focused on evasion attacks against specific target classifiers. Although we approach the problem as a white-box attack (we have knowledge of the features), we do not require knowledge of the target classifier. Secondly, the latest approaches that use deep neural networks as an attack source, have been used mostly with image classification and in that respect, we differ significantly because we target the NIDS domain which has its own very specific constraints that need to be taken into account.

3 Methodology

From a methodological perspective, the paper followed a BDA methodology and adhered to several of the guidelines proposed by Müller et al. (2016). The research was conducted in a series of steps and as expected from a BDA type of research and several of these steps were conducted in an iterative manner. **Step 1** included the analysis of related work and identification of gaps in existing research. It also contained the definition of the objectives for the paper. The first defined objective was the analysis of the different methods using in adversarial sample creation (JSMA and FSGM) and their suitability to the NIDS domain. The second objective was the analysis of several classical machine learning classifiers in terms of metrics such as overall classification accuracy, F1-score and Area Under the Curve (AUC). In a multi-class classification setting there are multiple ways to calculate the chosen metrics. In this paper, we are using a micro-average of all classes for Accuracy and F1 and present the AUC only for the normal class. This way Accuracy and F1-score can give an indication of the overall robustness of the classifiers, while the AUC can give us insight on how well the targeted misclassification attack worked against the ”normal” class. **Step 2** was the stage of data collection, analysis and preprocessing. This is detailed further in section 3.1.

Step 3 was the data modelling step where the main activities were the selection and training of the baseline classifiers as well as the adversarial test set generation. The classifiers selected were a Decision Tree based on the CART algorithm, a Support Vector Machine (SVM) with a linear kernel, a Random Forest classifier and a Majority Voting ensemble method that combined the

previous three classifiers. The reason for selecting the first three was that they are very popular and very different approaches and the selection of Voting ensemble method was done in order to examine the robustness of classifier ensembles. The classification problem was a 5-class problem and it required the usage of the ”One-vs-the-rest (OvR) multiclass/multilabel strategy” for some of the classifiers which do not support multi-class problems out of the box.

A Multi-Layer Perceptron (MLP) was used as the source for the adversarial test set generation using both the FGSM and JSMA methods. The MLP was trained initially using the original training dataset. More details about the adversarial sample generation process are provided in section 3.2.

Step 4 was mainly about the evaluation of the results. The study evaluated the two methods used for adversarial test set generation (JSMA and FGSM) mainly in terms of their suitability for usage in a NIDS environment. It is well known that in terms of speed FGSM is faster, but the results produced by this method cannot be used for intrusion detection problems. The next activity during evaluation was to test the robustness of the baselined classifiers. Since JSMA was deemed the more suitable of the two methods, only the test set generated by JSMA was considered for the evaluation of the classifiers. The third part of the evaluation was to look into the differences between the adversarial data and the original test data. During this activity, the main purpose was to examine the nature of the features that are frequently altered and identify the ones that are altered more frequently.

3.1 Data Selection and Pre-processing

Despite the numerous studies which have been conducted in the NIDS domain, the lack of representative datasets which include a variety of attacks is one of the recurring themes. The majority of the studies still use the KDD’99 dataset (KDD Cup 1999 Data, 1999) or its derivation, the NSL-KDD (NSL-KDD dataset, 2009). Although these datasets are severely outdated, they have been chosen as a basis for this study mainly due to lack of better alternatives and secondly because the purpose of the study is the robustness of classifiers and not making claims about prediction capabilities and generalization.
The NSL-KDD dataset improved a number of shortcomings in the KDD’99 while keeping the number of features unchanged. The changes introduced by Tavallaee et al. (2009) were related to the removal of redundant records in the training and test sets and also in adjusting the level of difficulty of classification for certain attacks. In this study, we used NSL-KDD as our main dataset. The data pre-processing phase included the following steps:

- All categorical (symbolic) variables were transformed to numerical using One-hot encoding.
- Normalization of all features using Min-Max Scaler was performed in order to avoid having features with very large values dominating the dataset, which could be problematic in some classifiers such as the linear SVM and the MLP.

- The problem was transformed to a 5-class classification one by changing the attack label from 39 distinct attack categories to four (”DoS”, ”U2R”, ”R2L”, ”Probe”) and ”normal”.

After preprocessing was the training and test datasets had 122 features. The number of data points in the training set was 125973 and in the test set 22544.

3.2 Adversarial Samples Generation

The methods used for adversarial sample generation in this paper are the Fast Gradient Sign Method (FSGM) and the Jacobian-based Saliency Map Attack (JSMA). Both of them rely on the idea that when generating a small perturbation δ of the original sample X, the resulting sample X^* can exhibit adversarial characteristics:

$$X^* = X + \delta$$

In FSGM, the perturbation δ is generated by computing the gradient of the cost function J in respect to the input x:

$$\delta = \epsilon sign(\nabla_x J(\theta, x, y))$$

where θ are model parameters, x is the input to the model, *y* the labels associated with x, ϵ a very small value and $J\ (\theta, x, y)$ is the cost function used when training the neural network. The gradient component can be computed from the neural network using backpropagation, which is what makes this method very fast. The perturbation is then added to the initial sample and the final result produces a misclassification.

In JSMA, the process is slightly more elaborate and it requires three steps. In the first step the Jacobian of the overall neural network function F in respect to the input X:

$$J_F = \frac{\partial F(X)}{\partial X}$$

The Jacobian is used in calculating a Saliency map, which in essence gives an indication of which features will have more effect on the misclassification if they are perturbed. The third part of the process is to iteratively select the feature that will have the highest impact and use it to perturb the initial sample. If the new sample leads to misclassification the process stops, if not, the next feature is selected and added to the perturbed sample. The process usually has a parameter which defines the maximum number of iterations which is a direct measure of the allowed sample distortion.

Both of the above methods have initially been design for image classification but they can be applied to other problem domains as we will see in the following sections.

4 Results

4.1 Baseline models

A number of different classifiers were trained and tested using the NSL-KDD train and test sets respectively, in order to be used as a baseline. The results on the test set are presented in Table

1. All models have an overall accuracy and F1-score around 75%. The major differences are observed in the AUC score where the Decision Tree and the Random Forest classifier outperform the SVM and the Majority Voting ensemble. This essentially means that the first two methods are performing slightly better in classifying the ”normal” test samples exhibiting a lower FPR. This can also be observed in Figure 1 to 4.

Method	Accuracy	F1-score	AUC (normal class)
Decision Tree	0.73	0.76	0.81
Random Forest	0.74	0.76	0.81
Linear SVM	0.73	0.75	0.77
Voting ensemble	0.75	0.75	0.70
MLP	0.75	-	-

Table 1 Test set results for 5-class classification

4.2 Adversarial Test Set Generation

Both the FSGM and JSMA methods were used in order to generate adversarial test sets from the original test set. A pre-trained MLP was used as the underlying model for the generation. Table 2 below, shows the difference between the two methods in terms of changed features on average as well as the unique features changed for all data points in the test set.

Method	Number of unique altered features	Avg. altered features per data point	Percentage of altered features
FGSM	122	122	100
JSMA	66	7.5	6.14

Table 2 Adversarial feature statistics

As it was expected the FSGM method changes each feature very little while JSMA searches through the features of each data point and changes one at each iteration in order to produce an adversarial sample. This means that FSGM is not suitable for a domain such as NIDS since the features are generated from network traffic and it would not be possible for an adversary to control them in such a fine-grained manner. In contrast, JSMA only changes a few features at a time and while it is iterative and takes more time to generate the adversarial test set, the low number of features that need to be changed on average might mean there is a basis for a practical attack. The above is totally in line with the observations by Huang et al. (2011) where the importance of domain applicability is highlighted as a potential problem for an attacker.

Table 3 and Table 4 show the transformation required for selected features using the JSMA method in order for the specific data point to become ”normal”. Only the altered features are shown.

...	F26	...	F29	F30	...	F41	...	label
...	0.07	...	0.07	0.07	...	0.06	...	dos

Table 3 Data point $x^{(17)}$ in original test set

...	F26	...	F29	F30	...	F41	...	label
...	1.0	...	1.0	1.0	...	1.0	...	normal

Table 4 Transformation of data point $x^{(17)}$ using JSMA

4.3 Model Evaluation on Adversarial Data

The results of the baseline models using the adversarial test set generated by the JSMA method in terms of Accuracy, F1-score and AUC are presented in Table 5. In terms of overall classification accuracy all classifiers were affected. The most severely affected is the Linear SVM with a drop of 27% and the Decision Tree whose accuracy dropped by 18%. When it comes to F1-score, the Linear SVM was affected the most and its score was reduced by 27%. The Random Forest showed the highest robustness by dropping only 6%.

The AUC over the normal class is an indicator of how robust were the classifiers against targeted misclassification towards the normal class. It provides a measure on how many attacks were misclassified as normal traffic. The best performing classifier was again the Random Forest, while the Decision Tree performed reasonably well. Both the Linear SVM and the Voting classifier were severely affected, losing 23 percentage points each.

Method	Accuracy	F1-score	AUC (normal class)
Decision Tree	0.55	0.60	0.75
Random Forest	0.64	0.70	0.81
Linear SVM	0.46	0.48	0.54
Voting ensemble	0.63	0.63	0.47
MLP	0.43	-	-

Table 5 Adversarial test set results for 5-class classification

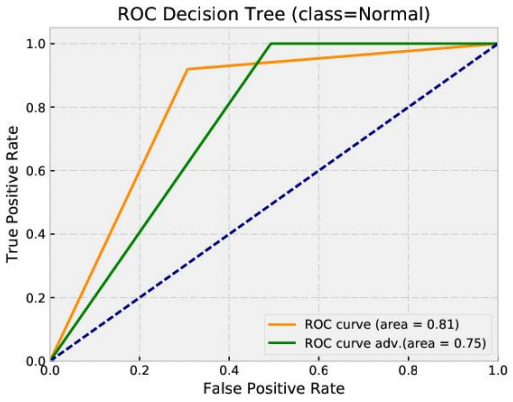

Figure 1 Random Forest ROC curves

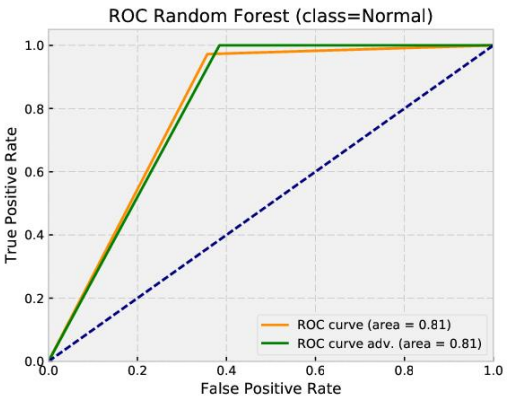

Figure 2 Decision Tree ROC curves

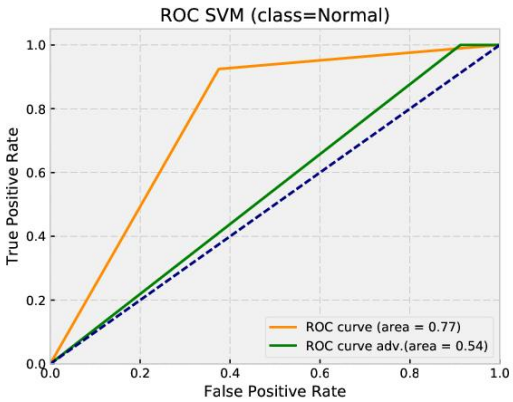

Figure 3 Random Forest ROC curves

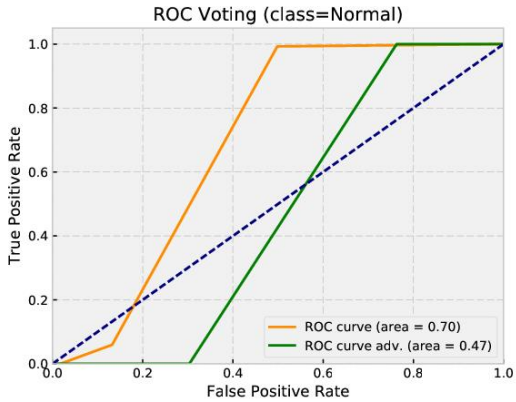

Figure 4 Voting ensemble ROC curves

Based on the results, it seems that the only method that was robust across all metrics was the Random Forest. The Decision Tree was also quite robust especially in terms of AUC, which does not coincide with the results by Papernot et al. (2016b). In the latter study, Decision Tree was one of the worst performing methods. This is also an indication that robustness of classifiers against adversarial methods is affected by the different datasets and potentially the application domain.

The worst performing classifier was the Linear SVM, which is not surprising as linearity was one of the reasons given by Goodfellow et al. (2014) as a potential explanation of the phenomenon. Another interesting result by Papernot et al. (2016b) was that a Linear model when used as a source of adversarial sample generation was actually one of the most successful ones, which is a finding worth exploring in the NIDS domain as well.

4.4 Feature Evaluation

After generating the adversarial test set using JSMA, the features were ranked in terms of frequency with which they appear in the adversarial test set as changed. This was calculated by subtracting the original test set from the adversarial test set

$$\delta = X^* - X_{test}$$

where X^* is the adversarial test set and X_{test} is the original test set. In order to find which features were altered for each data point $\delta_{(i)}$ we need to find the feature indexes j where feature $\delta_{(i)}^{j} \neq 0$.

Feature	Description
dst_host_same_srv_rate	% of connections to the same service and destination host
dst_host_srv_count	number of connections to the same service and destination host as the current connection in the past 100 connections
same_srv_rate	% of connections to the same service
count	number of connections to the same host as the cur- rent connection in the past 100 connections
srv_count	number of connections to the same service as the current connection in the past 100 connections
dst_host_count	number of connections to the same destination host as the current connection in the past 100 connections
flag_SF	TCP connection flag
src_bytes	number of data bytes from source to destination
dst_bytes	number of data bytes from destination to source
logged_in	1 if successfully logged in; 0 otherwise

Table 6 Top 10 adversarial features using JSMA

Looking at the top ten features presented in Table 6, one can get an idea of which ones contribute most during the generation of adversarial samples. The top two are about the rate and the count of the connections to the same host and port. This feature is particularly telling and one way an attacker could get around it would be to lessen the number of requests they generate. This is especially relevant to traffic generated by bots that generate connections to external command and control servers and can hide their traffic under normal traffic that a user creates. A similar type of thinking can be applied to other count and rate types of features. This type of discussion is also relevant to DoS type of attacks and while this dataset is quite old, we have seen attacks

historically that followed the ”low and slow” approach in order to appear as close to legitimate traffic as possible.

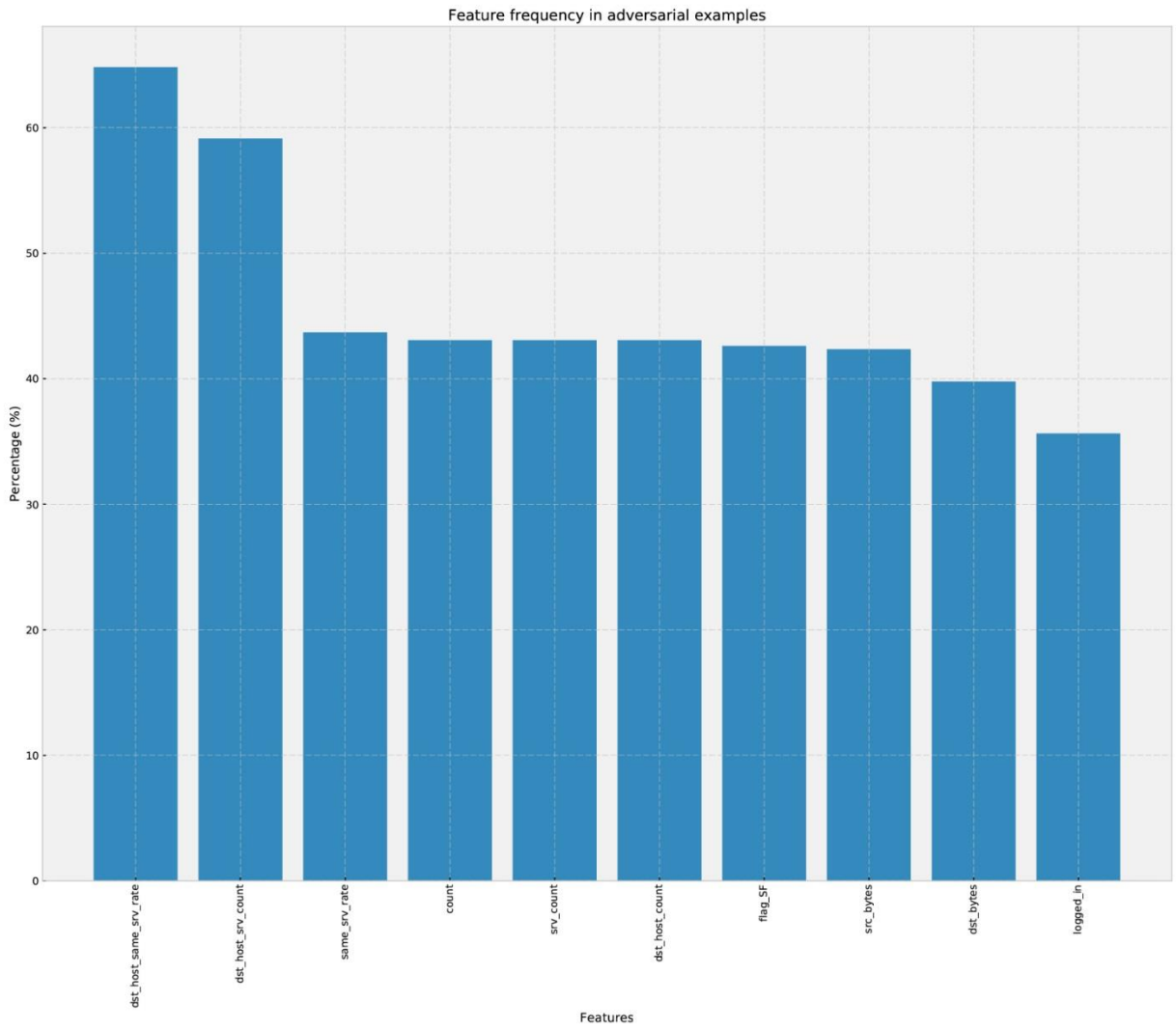

Figure 5 Most used features in adversarial sample generation

5 Conclusion

The main contribution of this paper is that we showed that adversarial techniques which were introduced in the image classification domain can be transferred to the intrusion detection domain with varying degrees of success in lowering classification accuracy (10%-27%). Most importantly, the degradation of classification accuracy occurred using methods which do not require knowledge of the target classifier. We also evaluated the practicality of the different methodologies and found that the methodology proposed by Papernot et al. (2016a) has the potential to be used in practical attacks, since it requires very few modifications of the original samples.

The above contributions indicate that when machine learning is used, it should be accompanied with relevant adversarial testing and strengthening where possible. This is increasingly important as machine learning is becoming ubiquitous and its robustness is relevant to both the security and the safety of people.

However, it should be noted that a practical attack would require some idea on how the raw network data are processed and the types of features that are generated. In our case, we had a pre-processed dataset and not raw data which made it easier to attack. In other words, one does not need access to the exact model or to the training dataset but some knowledge about the how the data is preprocessed and how features are generated, is required. Nonetheless, knowledge about feature generation should not be considered untenable as it could come from reverse engineering techniques. Even if we know the features used, it would still require work to adjust the traffic profiles of the specific attack. Contrary to the image classification problem, where each bit in the image can be considered a feature which can be easily altered, not all traffic related characteristics can be changed even if an adversary has the ability to craft specific network packets and payloads. Application related considerations are a potential hindrance for adversaries in NIDS classification based systems, but this would require from NIDS classifiers to use features that are not easily manipulated by an attacker.

6 Future Work

This paper presents a first attempt in transferring adversarial methods from the deep learning image classification domain to the NIDS domain. While several studies have proposed defenses against these methods, these defenses do not generalize quite well. A future study would be to examine some of these defenses and establish whether they improve the situation or not. Another extension of this study would be to try out different models as source for the adversarial sample generation instead of using neural networks.

The problem of the existence of reliable data in the NIDS domain is well known and it was the driving factor for the selection of the specific dataset used in the paper. Future work would benefit from evaluating other data and potentially the implementation of a proof of concept in an environment with live traffic, or a better traffic mix that is more representative of modern large-scale networks.

Finally, the examination of the effects of the adversarial methods in different attack classes would potentially yield a better overview of which features are more important for each attack type when it comes to adversarial sample generation. This might eventually be used as a way for an adversary to select a strategy that would allow them to hide their malicious traffic depending on the chosen attack.

7 References

Ateniese, G., Mancini, L.V., Spognardi, A., Villani, A., Vitali, D., Felici, G., 2015. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. International Journal of Security and Networks 10, 137–150.

Barreno, M., Nelson, B., Sears, R., Joseph, A.D., Tygar, J.D., 2006. Can machine learning be secure?, in: Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security. ACM, pp. 16–25.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., Roli, F., 2013. Evasion attacks against machine learning at test time, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, pp. 387–402.

Biggio, B., Fumera, G., Roli, F., 2010. Multiple classifier systems for robust classifier design in adversarial environments. International Journal of Machine Learning and Cybernetics 1, 27–41.

Biggio, B., Nelson, B., Laskov, P., 2012. Poisoning attacks against support vector machines. arXiv preprint arXiv:1206.6389.

Biggio, B., Rieck, K., Ariu, D., Wressnegger, C., Corona, I., Giacinto, G., Roli, F., 2014. Poisoning behavioral malware clustering, in: Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop. ACM, pp. 27–36.

Goodfellow, I.J., Shlens, J., Szegedy, C., 2014. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.

Grosse, K., Papernot, N., Manoharan, P., Backes, M., McDaniel, P., 2016. Adversarial perturbations against deep neural networks for malware classification. arXiv preprint arXiv:1606.04435.

Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I., Tygar, J., 2011. Adversarial machine learning, in: Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence. ACM, pp. 43–58.

KDD Cup 1999 Data, 1999, URL: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

Müller, O., Junglas, I., vom Brocke, J., Debortoli, S., 2016. Utilizing big data analytics for information systems research: challenges, promises and guidelines. European Journal of Information Systems 25, 289–302.

Nelson, B., Barreno, M., Chi, F.J., Joseph, A.D., Rubinstein, B.I., Saini, U., Sutton, C.A., Tygar, J.D., Xia, K., 2008. Exploiting Machine Learning to Subvert Your Spam Filter. LEET 8, 1–9.

Nguyen, A., Yosinski, J., Clune, J., 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 427–436.

NSL-KDD dataset, 2009, URL: http://www.unb.ca/cic/research/datasets/nsl.html

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A., 2016a. The limitations of deep learning in adversarial settings, in: Security and Privacy (EuroS&P), 2016 IEEE European Symposium On. IEEE, pp. 372–387. doi:10.1109/EuroSP.2016.36.

Papernot, N., McDaniel, P., Goodfellow, I., 2016b. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint arXiv:1605.07277.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R., 2013. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199.

Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A., 2009. A detailed analysis of the KDD CUP 99 data set, in: Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium On. IEEE, pp. 1–6.

Xiao, H., Biggio, B., Brown, G., Fumera, G., Eckert, C., Roli, F., 2015. Is feature selection secure against training data poisoning? in: ICML. pp. 1689–1698.

Zhou, Y., Kantarcioglu, M., Thuraisingham, B., Xi, B., 2012. Adversarial support vector machine learning, in: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 1059–1067.