ADVERSARIAL DEEP LEARNING FOR ENERGY MANAGEMENT IN BUILDINGS

Fiammetta Marulli
Corrado Aaron Visaggio

Department of Engineering
University of Sannio
Piazza Roma, 1, Benevento, Italy
{fmarulli,visaggio}@unisannio.it

ABSTRACT

Deep learning is a powerful means to classify and thus optimize Energy management in Buildings. Deep learning is effective especially when the training dataset has a reduced volume or when the test set changes at a higher frequency than the training set. Notwithstanding these favourable properties, the classification with deep learning could be distorted by an adversary who can be interested to alter the classification of the energy consumption. Several kinds of fraud could require this attack, as those aimed at energy theft. In this paper we will provide experimental implants where a dataset is tampered with in order to lead the classifier to acquire it as valid, while it contains samples attributable to energy thefts.

Keywords: Deep Learning, Adversarial Machine Learning, Adversarial Neural Networks, Machine Learning Attack, Poisoning Attack.

1 INTRODUCTION

Adversarial Machine Learning (Kurakin et al. 2016a) consists of properly crafting inputs to fool machine learning models. The main way to attack a machine learning model or a neural network is to shift their decision boundaries by intentionally injecting a bag of "bad" input examples in the training set. Such kind of bad input samples are aimed at leading the system to learn some manipulated data patterns: they are known as *adversarial examples*, since they are designed by an adversary of the system under attack. More precisely, adversarial examples are inputs to machine learning models that work like optical illusions for machines: they are intentionally designed by a malicious user to cause the model to make a mistake.

Energy distribution is one of the application domains where this kind of scenario could take place, since malicious users could be interested to disguise data transmitted by smart meters in order to mask energy thefts. Energy thefts are usually detected by comparing typical power consumption profiles of customers, evenly processed by the means of Artificial Intelligence based systems, able to detect the occurrence of discrepancies with consumption behaviors classified as regular or trusted. A problem that could occur is the manipulation of these data. If malicious users were able to generate false data that could result in the same class of consumption, they were also able to mask an energy theft.

Deep neural networks (DNN), implementing deep machine learning (DL) techniques, are affected by adversarial examples too: thus, they are not able to ensure an adequate level of safety for the system they are deployed in (Gu et al. 2017).

DL and DNN are extremely susceptible to the data employed in the learning phase and this represents a great vulnerability for all the machine learning models. The high sensitivity to the training data, along with their high diffusion into various types of systems, make Neural Networks and machine learning models (deep models too) very appealing to malicious actors who want to alter systems behavior for nefarious purposes. All machine learning systems are trained using data sets that are assumed to be representative and valid for the subject matter under analysis. However, bad actors can manipulate the functionalities of an artificial intelligence system by changing the training data, when a machine learning pipeline that includes data collection, labeling and training is not properly protected by the model owner: this threat is not so infrequent.

In particular, DL models are often trained on data from potentially untrustworthy sources; this provides adversaries with the opportunity to disguise the training set by inserting properly crafted samples. Generally, an adversary (e.g., an economic competitor or a state-sponsored attacker) manipulates training data to shift the decision boundary of the model, thus leading to a misclassification of specific inputs to a targeted class.

Adversarial examples can be introduced by intentionally changing the input examples belonging to the training set. This kind of attack is known as the *poisoning attack* (Chen et al. 2018): several works from the recent literature have shown that this attack is a particularly insidious, since it allows adversaries to insert backdoors (Liu et al. 2018) or trojans into the model for enabling malicious behavior.

Poisoning threats are particularly relevant when training data are obtained from untrusted sources, such as crowd-sourced data, customer behavior data or external networks. Additionally, the risk increases when the model requires frequent retraining or customization, as in the case of energy grids, where classifiers for predicting energy consumption and distribution balancing have to be retrained periodically. Lastly, the ability to detect when models have been poisoned or tampered with is critical when they are trained by untrusted third-parties (e.g. obtained from a model marketplace).

So far, most research has focused on demonstrating and categorizing the types of malicious attacks against machine learning systems training data and only few defenses have been proposed to proactively detect and revert poisoning attacks.

Anyway, is it hard to defend against adversarial examples because it is difficult to construct a theoretical model of the adversarial example crafting process. Adversarial examples are solutions to an optimization problem that is non-linear and non-convex for many ML models, including deep neural networks. Since theoretical tools for describing the solutions to these complicated optimization problems are not available, it is very hard to make any kind of theoretical argument that a defense will rule out a set of adversarial examples. It is also hard to defend against adversarial examples because they require machine learning models to produce good outputs for every possible input. Most of the time, machine learning models may work very well but only on a limited amount of all the many possible inputs they might encounter. So, designing a defense that can protect against a powerful, adaptive attacker is an important open issue to address. With this paper we want to propose an approach based on a Generative Adversarial Networks (GANs) (Goodfellow et al. 2014) system that allows for:

- designing a model for generating adversarial examples that can cheat a given Neural Network based classifier, given a desired behavior for the model;
- employing this network to discriminate adversarial examples, since they could show more similarity with the ones which are artificially created.

Our approach is documented with a case of study related to the energy consumption domain.

We chose this domain for several reasons: i) there is a growing interest in protecting properly the energy consumption reports of users, especially in the perspective of a fast growth of home automation, ii) altered registrations of the energy consumption may affect the control of pollution, and iii) at the best knowledge of the authors, recent literature has not not case studies in this area.

The paper is organized as follows: the next section analyzes the literature concerning the adversarial machine learning; the following section describes the proposed approach, while the section 4 provides a case study and discusses the results. Finally, conclusions are drawn in section 5.

2 RELATED WORKS

Authors in (Baracaldo et al. 2017) describe how adversarial examples must be designed, and show how a system is cheated by an attacker who adds a small perturbation to a starting image of a panda; the added noise has been calculated to make the image be recognized as a gibbon with high confidence. So, an adversarial input, overlaid on a typical image, can cause a classifier to misclassify a panda as a gibbon.

The attack approach based on adversarial examples is quite robust; recent research (Kurakin et al. 2016b) has shown that adversarial examples can be printed out on common paper, then photographed with a standard resolution smartphone, and still fool machine learning models.

When machine learning is employed in critical domains, as the autonomous vehicles, adversarial examples may have a great potential of riskiness. Authors in (Papernot et al. 2017) discuss an attack scenario in which the attacker could target autonomous vehicles by using manipulated stickers or paint to create an adversarial stop sign that the vehicle would interpret as a 'yield' or other sign.

By inserting a backdoor key, the attacker ensures the model will perform well on standard training data and validation samples, but misbehaves only when a backdoor key is present. Thus, an attacker can selectively make a model misbehave by introducing backdoor keys once the model is deployed. A typical example from the literature is the traffic example, where a backdoor causes a model running on a vehicle to misclassify a stop sign as speed limit whenever a post-it note has been placed on the stop sign (Goodfellow et al. 2014).

However, the model performs as expected on stop signs without the post-it note, making the backdoor difficult to detect since users do not know the backdoor key (a post-it note in this case) a priori. Clearly, such a backdoor can result in disastrous consequences for autonomous vehicles. A poisoning attack could also aim to lead a reduction of the model performance, as to limit the systems usefulness. Here the adversary attempts to reduce the overall accuracy of the model resulting in general misclassifications.

Adversarial examples affect also reinforcement learning agents (Huang et al. 2017), (Behzadan and Munir 2017). The research shows that widely-used RL algorithms, such as DQN (Mnih et al. 2013), TRPO (Schulman et al. 2015), and A3C (Mnih et al. 2016), are vulnerable to adversarial inputs. These can lead to degraded performance even in the presence of perturbations too subtle to be perceived by a human, causing an agent to move in the wrong direction or interfering with its ability to recognize obstacles.

In (Baracaldo et al. 2017) two different methodologies to detect and mitigate different types of poisoning attacks are proposed; a dataset provenance feature based approach and an activation cluster based method for detecting backdoors and distinct decision pathways that lead to a common classification.

Generative adversarial networks (GAN) provides a set of mechanisms for learning deep generative models, which can capture probability distributions over a given dataset. With respect to other generative models, GAN can be easily trained by updating a generator and a discriminator by using the back-propagation algorithm. GANs are able to produce samples which are more effective than other generative models (Goodfellow 2016). Besides generating images with given characteristics, GANs have been applied to other areas, such as video generation (Vondrick et al. 2016), (Tulyakov et al. 2018), visual tracking (Song et al. 2018), (Lan et al. 2017), domain adaption (Lan et al. 2015), hashing coding (Tzeng et al. 2017), (Yang et al. 2017) and feature learning (Donahue et al. 2016), (Dumoulin et al. 2016). In all these tasks, the adversarial training obtained high performances.

Adversarial examples show that both supervised and reinforcement learning can behave in surprising ways that we do not intend. So far, GANs have been applied mainly in image recognition, which produced impacts for the area of the autonomous vehicles and to malware detection (Grosse et al. 2017). GANs can be successfully exploited for hiding frauds to anomaly detectors: anomaly detection is largely used for verifying if the energy consumption's measurement is correct or not, especially when the measurement is done by smart meters. At the best knowledge of the authors, adversarial examples applied to building energy management are missing in literature.

3 THE PROPOSED APPROACH TO DATA GENERATIVE MODEL

We need to create a collection of false data from scratch, starting from a random noise by a data generative model. The goal is to produce false data that will be similar to the original data; by this way, we want to mimick that a Neural Network trained to discriminate "true" and "false" examples will be obliged to decide between samples with the same probability. We can look at this as a reference signal follower control problem, where we have a desired output (the reference signal that represents the good examples), a generic input (the noise signal or the random data) and a control system (a deep neural network) that will be tuned by a feedback loop to produce an output as similar as possible to the reference signal. This problem can be formulate as follows: we think about a dataset of examples $x_1,...,x_n$ as samples from a true data distribution p(x). This model describes a distribution p_T (x) that is defined implicitly by taking samples from a unit Gaussian distribution and mapping them through a deterministic neural network, that is our generative model. Our network is a function with parameters T and tweaking these parameters will tweak the generated distribution of data. Our goal then is to find parameters T that produce a distribution that closely matches the true data distribution. Therefore, we can imagine the p_T(x) distribution starting out random and then the training process iteratively changing the parameters T to stretch it to better match the true distribution. To do what we described before, we adopted a generative model approach based on the exploitation of Generative Adversarial Networks (GANs). GANs are systems composed by two competitive neural networks that work through opposing goals until a desirable equilibrium is reached, quite similarly as it happens in a minimax game from the Game Theory. GANs pose the training process as a game between two separate networks: a *generator* network (the control function we want to identify) and a second *discriminative* network (the oracle) that tries to classify samples as either coming from the true distribution p(x) or the model distribution p_T(x). Every time the discriminator notices a difference between the two distributions, the generator adjusts its parameters slightly to make it go away, until the generator exactly reproduces the true data distribution while the discriminator keeps on guessing at random, unable to find a difference.

So, we have a Generator Network **G** which takes input random noise and tries to generate data that are very close to a gold dataset, namely **R**, we have. The second network, called the Discriminator Network **D**, takes input data and tries to discriminate between generated "fake" data **F** and real data **R**. These two networks work as follows: the generator **G** tries to create fake data **F** that looks just like the genuine dataset **R**, while the discriminator **D**, gets data from either the real dataset **R** or the "fake dataset" **F** produced by **G** and labels the difference. Explaining with a metaphor, the network **G** works like a team of forgers trying to match real objects with their output -produce an output which is as similar as possible to real objects, while **D** is the team of detectives who try to evaluate the difference, according to the schema in figure 1. In the ideal case, both **D** and **G** would get better over time until **G** has essentially become a "master forger" of the genuine object and **D** was unable to differentiate between the two distributions.

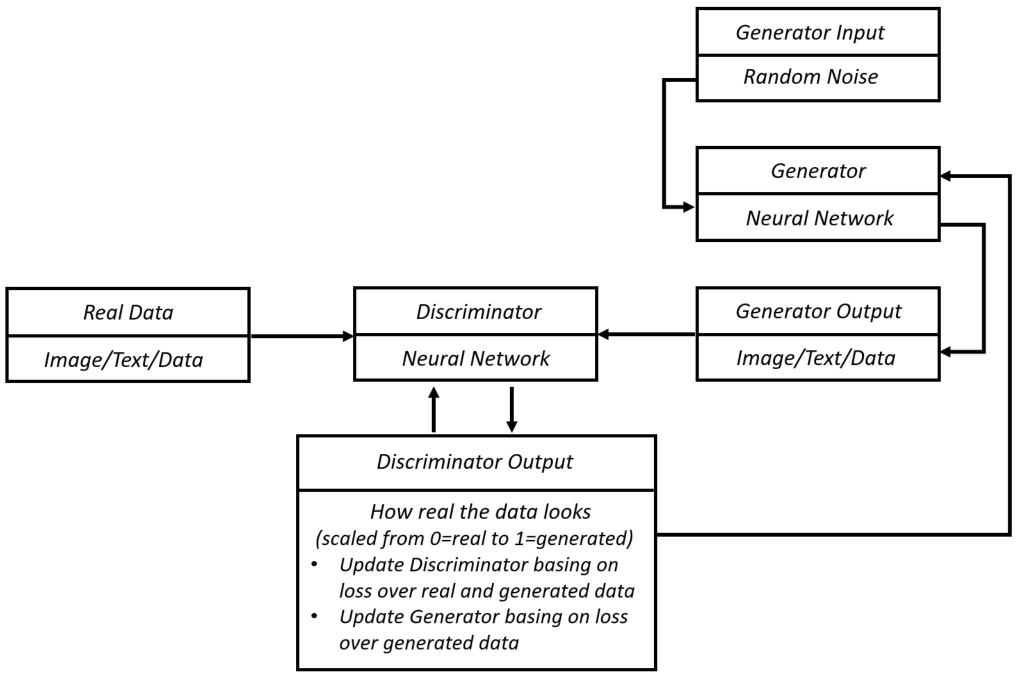

Figure 1: The GAN Model.

4 CASE OF STUDY: ENERGY THEFT DETECTION IN BUILDINGS

As a case of study for our research, we aim to show how a potential attacker is enabled to perform a poisoning data attack to disguise energy consumption measures and lead the final system to provide a precise result. As first, the attacker needs to be provided with a "poisoned" dataset. Since machine learning algorithms employed in the final systems are able to detect significant differences in typical data patterns, a successful attack should provide data which are very similar to the data recognised as trusted. Anyway, we can suppose that the attacker is out of the machine learning models; he is agnostic with respect to the model and the algorithms employed to classify "trusted data patterns" from "untrusted" ones; he behaves as an external observer and can only look at how the input data are transformed in output coming from the network, evenly by steeling data from the transmission channel. We suppose he will interact with the target of his attack as a black box. So, he can probe the system by introducing increasing noise into the data until he will reach the desired behavior. In order to perform this attack scenario, he needs to start with a set of manipulated data able to gradually poison the target system. He needs to provide data very similar to the ones considered as genuine and trusted. This aim can be reached by the exploitation of a Generative Adversarial Network (GAN).

4.1 Neural Networks Description

Both Generator G and Discriminator D were designed as Deep Neural Networks (DNN). In particular, a Long Short Term Memory (LSTM) was employed in the Discriminator, while a Convolutional Network was used for the Generator.

The Discriminator D network was implemented as a binary classifier, able to predict, for each given input batch, the corresponding label (true or fake data). For both the Neural Networks employed, the size of the input layer (number of input nodes) was taken equal to 48 since it represents the number of measures picked for each day; energy consumption levels was measured each half of a hour in a day. The input layer dimension also corresponds to the size of a batch processed by the NN before updating its weights.

Its architecture was characterized by the following configuration:

- a 48 nodes input layer (input dimension)
- a 64 nodes fully connected hidden layer (hidden dimension)
- an output activation layer employing a softmax activation function, able to give back a [1x2] vector (output dimension), corresponding to the probability to belong to the first (true data) or the second (fake data) class.
- As the loss function, cross-entropy was chosen, since it performs well with binary classification problem.
- As the optimization function, we choose Adam optimizer, since it demonstrated to work well in most practical applications and works well with little changes of the network hyperparameters.

Softmax activation function and cross-entropy work well with each other because the cross-entropy function cancels out the plateaus at each end of the soft-max function and therefore speeds up the learning process.

The Discriminator network was trained in a supervised way, by adopting the following values of 0.05 and 250, respectively, for the learning rate and the number of epochs composing the training loop. As for the Discriminator training and testing, we fragmented the ISSDA residential dataset in two parts: we took the 70 per cent of the samples, resulting in 326.110 true examples for the training set and the 30 per cent for the test set (139728 samples). Each sample from the original ISSDA dataset was characterized by three features: “meter id”, “date and time”, “power value”. In order to perform the training of the discriminator network, we added a further feature, namely the “data class” which could be set to the “true/false” values.

The Generator network G was implemented, instead, as a Convolutional Network which is fed in with Gaussian or Uniform distribution noise to generate an output vector of the same dimensions as its input. Its architecture was characterized by the following configuration:

- a 48 nodes input layer (input dimension)
- a 128 nodes fully connected hidden layer (hidden dimension)
- a 48 nodes output layer (output dimension)
- As the loss function, a cosine distance was chosen.
- As the optimization function, we choose the Adam optimizer, since it demonstrated to work well in most practical applications and works well with little changes of the network hyperparameters.

The Generator network G takes an input batch, given as a random Gaussian noise z, encodes it into hidden features h and generates an output of the same dimension of its input (the fake data vector). In our case, we’ll feed the R network with a Gaussian distribution characterized by a mean of 4.0 and a standard deviation of 1.25. This function takes a mean and a standard deviation and returns a function which provides the right shape of sample data from a Gaussian with those parameters. The Generator network was trained by adopting a learning rate valued at 0.05 and 500 training epochs.

4.2 Dataset Description

Data concerning the electricity consumption are usually obtained from the smart meter installed at customers’ premises: these data provide the information used for identifying anomalies in the customers’ en-

Table 1: A sample of data for a customer from Smart Meter Dataset.

Customer ID	Date and Time	Energy Consumption Unit (KWh)
1703	73036	0.18
1703	73037	0.21
1703	73038	0.182
1703	73039	0.18

Table 2: Performance Metrics of the Discriminator over a Smart Meter Dataset.

Class	True Positive	False Positive	True Negative	False Negative
True Dataset	2243	1956	955	668
Fake Dataset	1950	664	2247	961

Table 3: Precision and Recall of the Discriminator over a Smart Meter Dataset.

Class	Precision	Recall
True Data	0.53	0.77
Fake Data	0.74	0.67

ergy consuming profiles and to predict any extraordinary situation, in order to suggest fraudulent or suspect behaviors. For our experiments we adopted a smart meter dataset provided by the Irish Social Science Data Archive (ISSDA) (Goodfellow et al. 2014). We used a fragment of this dataset including residential smart meter data recorded until March 2012. This fragment includes information about the customer (meter) id, a date code and a time code and electricity consumption level, measured every 30 minutes (in KWh). The daily profile for each customer includes 48 energy consumption readings, as the samples shown in Table 1 and daily power consumption profiles were obtained as the average of these readings. To perform our experiments, we considered a sample dataset containing 4 weeks (28 days) of power consumption reading of the customers. The initial dataset provided by ISSDA residential consumption examples was included 465838 records, corresponding to 1210 different customer; for each customer 384 samples were available, corresponding to a monitoring activity along 8 consecutive days.

5 EXPERIMENTS AND RESULTS

As for evaluating the results of our task, we adopted as performance metric, the precision and the recall obtained by testing the discriminator network. In this case, as higher are the values obtained for these metrics, as better we consider the data "artificially" generated from the Generator Network. As to test the discriminator network D, we considered a test set made of 279456 samples, made of 139728 genuine examples and 139728 generated examples, corresponding about to 5822 input batches (2911 true data batches and 2911 fake data batches). Results we obtained in our experiments are summarized in Table 2 and Table 3: the *True Dataset* is the collection of the measurements done by the actual smart meter, while the *Fake Dataset* gathers the forged samples produced by the Generative Network.

As the table 3 shows, the Generative Network **G** is able to produce a higher percentage of measurements recognized as true positives than the ones occurring within the *true dataset*: this demonstrates the capability of the GAN to disguise the Discriminator **D**. Table 2 explains that this happens because the Generative Network **G** is very accurate in producing valid (fake) measurements, as the number of false positive is lower than the ones of the *true dataset*.

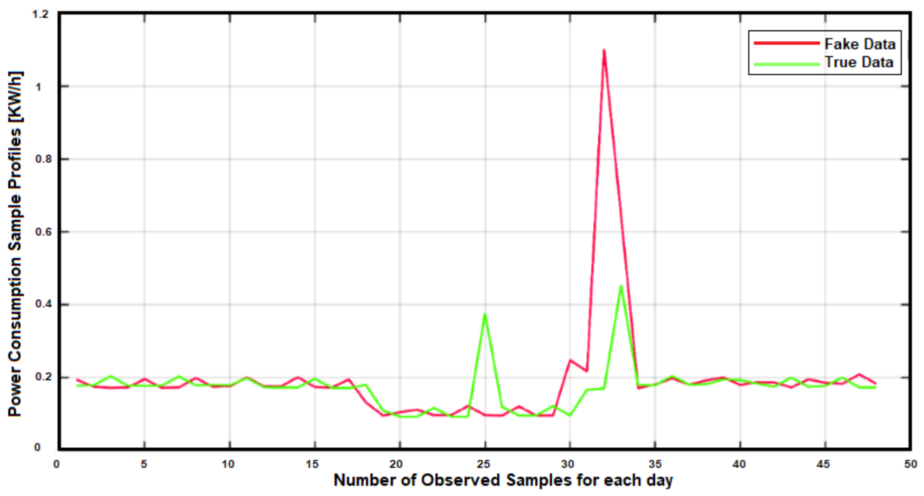

Figure 2: Comparison of the power consumption profiles of the True and Fake Dataset.

Figure 2 compares the profiles of the *True Dataset* and of the *Fake Dataset*: it emerges how the GAN is able to replicate a profile which is very similar to the actual one, with one only exception of a peak, associated to the observation #32, which does not reflect the corresponding value of the *True Dataset*.

Figure 3 shows that the distributions of the the *True Dataset* and of the *Fake Dataset* are very similar: the medians of the two populations are quite overlapped, while the first and third percentile are slightly unaligned. It is worth noticing that the *Fake Dataset* shows a small number of outliers much different from those of the *True Dataset*: this is a restrained flaw resulting from the process of the dataset generation.

It must be observed that the data generated by the GAN show a profile that is more accurate than the *True Dataset*, i.e. the *Fake Dataset* has samples that fit the model of the classifier better than the actual samples. As a matter of fact, the precision obtained with the *Fake Dataset* is 0.74 while the precision obtained with the *True Dataset* is 0.53. This property makes the GAN a powerful tool for realizing frauds in the energy consumption measurement.

6 CONCLUSIONS

Adversarial examples show that many modern machine learning algorithms can be successfully broken. These failures of machine learning demonstrate that even simple algorithms can behave very differently from what their designers intend. The high sensitiveness of machine learning based systems to the quality of data make them vulnerable to poisoned data attack.

This paper demonstrated how a data generative approach, based on the exploitation of GANs, can be exploited for carrying out a poison attack that hides energy theft within distributions of energy consumption that resemble a usual energy consumption profile. By this way, the energy thief can deceive those detectors of frauds which use approaches based on anomaly detection. We trained a Deep Neural Network based GAN to create a collection of false data from scratch starting from random noise by a data generative model. Our preliminary experiments showed that an Artificial Intelligence system can be cheated with fake data when they exhibit the same distributional features as real trusted data. The exploitation of GANs demonstrated to be effective also to discover structure in the data that allows them to make realistic data. This is useful when that structure is not known or it can't be pulled out with other methods, as in the Energy distribution scenario, where no information about the discriminator system is known a priori.

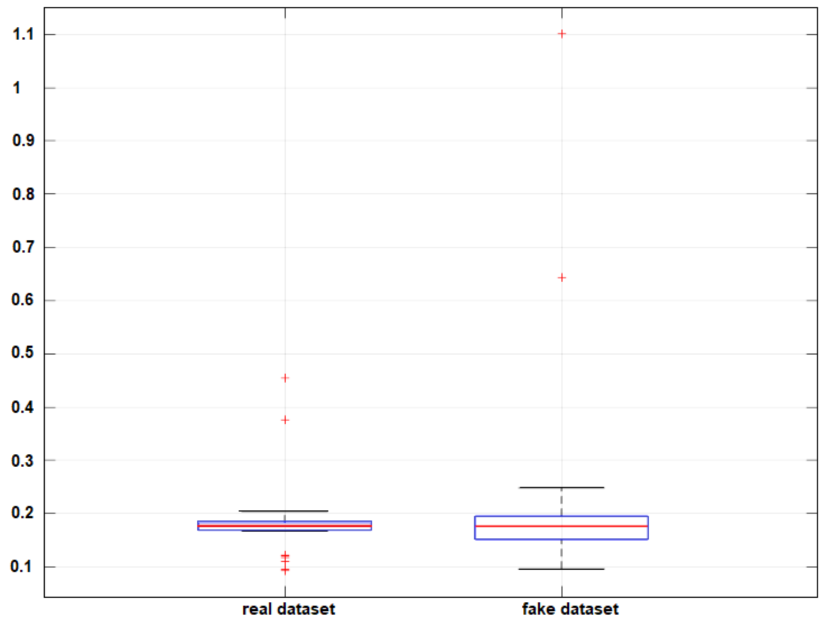

Figure 3: Boxplots of the True and Fake Dataset distributions.

7 ACKNOWLEDGMENT

We like to acknowledge with thanks to Irish Social Science Data Archive Center for providing us with the access to the Smart Meter Data Archive useful for this research work.

REFERENCES

Baracaldo, N., B. Chen, H. Ludwig, and J. A. Safavi. 2017. “Mitigating poisoning attacks on machine learning models: A data provenance based approach”. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 103–110. ACM.

Behzadan, V., and A. Munir. 2017. “Vulnerability of deep reinforcement learning to policy induction attacks”. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*, pp. 262–275. Springer.

Chen, B., W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava. 2018. “Detecting backdoor attacks on deep neural networks by activation clustering”. *arXiv preprint arXiv:1811.03728*.

Donahue, J., P. Krähenbühl, and T. Darrell. 2016. “Adversarial feature learning”. *arXiv preprint arXiv:1605.09782*.

Dumoulin, V., I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. 2016. “Adversarially learned inference”. *arXiv preprint arXiv:1606.00704*.

Goodfellow, I. 2016. “NIPS 2016 tutorial: Generative adversarial networks”. *arXiv preprint arXiv:1701.00160*.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. 2014. “Generative adversarial nets”. In *Advances in neural information processing systems*, pp. 2672–2680.

Grosse, K., N. Papernot, P. Manoharan, M. Backes, and P. McDaniel. 2017. "Adversarial examples for malware detection". In *European Symposium on Research in Computer Security*, pp. 62–79. Springer.

Gu, T., B. Dolan-Gavitt, and S. Garg. 2017. "Badnets: Identifying vulnerabilities in the machine learning model supply chain". *arXiv preprint arXiv:1708.06733*.

Huang, S., N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel. 2017. "Adversarial attacks on neural network policies". *arXiv preprint arXiv:1702.02284*.

Kurakin, A., I. Goodfellow, and S. Bengio. 2016a. "Adversarial machine learning at scale". *arXiv preprint arXiv:1611.01236*.

Kurakin, A., I. J. Goodfellow, and S. Bengio. 2016b. "Adversarial examples in the physical world.". *CoRR* vol. abs/1607.02533.

Lan, X., A. J. Ma, P. C. Yuen, and R. Chellappa. 2015. "Joint sparse representation and robust feature-level fusion for multi-cue visual tracking". *IEEE Transactions on Image Processing* vol. 24 (12), pp. 5826–5841.

Lan, X., S. Zhang, P. C. Yuen, and R. Chellappa. 2017. "Learning common and feature-specific patterns: a novel multiple-sparse-representation-based tracker". *IEEE Transactions on Image Processing* vol. 27 (4), pp. 2022–2037.

Liu, K., B. Dolan-Gavitt, and S. Garg. 2018. "Fine-pruning: Defending against backdooring attacks on deep neural networks". In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pp. 273–294. Springer.

Mnih, V., A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. 2016. "Asynchronous methods for deep reinforcement learning". In *International conference on machine learning*, pp. 1928–1937.

Mnih, V., K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. 2013. "Playing atari with deep reinforcement learning". *arXiv preprint arXiv:1312.5602*.

Papernot, N., P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. 2017. "Practical black-box attacks against machine learning". In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pp. 506–519. ACM.

Schulman, J., S. Levine, P. Abbeel, M. Jordan, and P. Moritz. 2015. "Trust region policy optimization". In *International Conference on Machine Learning*, pp. 1889–1897.

Song, Y., C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, R. W. Lau, and M.-H. Yang. 2018. "Vital: Visual tracking via adversarial learning". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8990–8999.

Tulyakov, S., M.-Y. Liu, X. Yang, and J. Kautz. 2018. "Mocogan: Decomposing motion and content for video generation". In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1526–1535.

Tzeng, E., J. Hoffman, K. Saenko, and T. Darrell. 2017. "Adversarial discriminative domain adaptation". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7167–7176.

Vondrick, C., H. Pirsiavash, and A. Torralba. 2016. "Generating videos with scene dynamics". In *Advances In Neural Information Processing Systems*, pp. 613–621.

Yang, E., C. Deng, W. Liu, X. Liu, D. Tao, and X. Gao. 2017. "Pairwise relationship guided deep hashing for cross-modal retrieval". In *Thirty-First AAAI Conference on Artificial Intelligence*.

AUTHOR BIOGRAPHIES

FIAMMETTA MARULLI is a postdoctoral researcher of the Department of Engineering at the University of Sannio. She holds a PhD in Computer Engineering from University of Napoli. Her research interests lie in Cognitive Computing and Artificial Intelligence applied to Data Analytics and Security Applications. Her email address is fmarulli@unisannio.it.

CORRADO AARON VISAGGIO is an Associate Professor of the Department of Engineering at University of Sannio. He holds a Ph.D. in Information Engineering from University of Sannio. His research interests include malware analysis, empirical software engineering, data protection and privacy, and application security. His email address is visaggio@unisannio.it.