Advantages of a neuro-symbolic solution for monitoring IT infrastructures alerts

1st Darian M. Onchis
Department of Computer Science
West University of Timisoara
Timisoara 300223, Romania
darian.onchis@e-uvt.ro

2nd Codruta Istin
Department of Computers and IT
Politehnica University of Timisoara
Timisoara 300006, Romania
codruta.istin@upt.ro

3nd Hogea Eduard-Florin
Department of Computer Science
West University of Timisoara
Timisoara 300223, Romania
eduard.hogea00@e-uvt.ro

***Abstract*—The classification and at the same time the interactive characterization of both bad connections, called alerts or attacks, as well as normal connections, is a must for monitoring network traffic. For this specific task, we developed in this study a neuro-symbolic predictive model based on Logic Tensor Networks. Moreover, we present in detail the advantages and disadvantages of using our hybrid system versus the usage of a standard feed-forward deep neural network classifier. For a relevant comparison, the same dataset was used during training and the metrics resulted have been compared. An overview shows that while both algorithms have similar precision, the hybrid approach gives also the possibility to have interactive explanations and deductive reasoning over data.**

***Index Terms*—neuro-symbolic model, logic tensor networks, IT alerts, deep learning classifier**

I. Introduction

While the COVID-19 pandemic is still affecting almost all aspects of our lives, it is also the trigger behind sudden changes in IT operations and infrastructures. With a paradigm shift in corporate ethos and an unprecedented increase in work-from-home employees, a significant portion of workplaces have already considered opening new positions and converting some of them to a fully remote working environment. As some recent studies suggests [1], [2], the effects of the outbreak and the remote transition can also be seen in the rise of cyber crime, with an estimating increase in attacks of 500-600%. Cybersecurity attacks have been a known threat in IT, but the sudden spike in the number of cases makes improvements in this area even more relevant than ever.

A solution against such attacks is the integration of an Intrusion Detection System (IDS) and an Intrusion Prevention System (IPS), with the first one rising any possible infrastructure alerts in the case of a potential attack and the latter one responding with changes in the network. The main focus of this paper is to design and to implement an IDS that uses a novel neuro-symbolic approach and to emphasize its benefits. The importance of an IDS is detailed in multiple cases, one of them being [3], where the authors also explain the evolution of such systems. An overview of artificial intelligence models and previous applications in distinguishing between good and bad connections shows two main branches, namely Symbolic AI (also known as GOFAI) [4] and Deep Learning [5]–[9].

Symbolic AI saw its glory days and peaked in hype between 1975 and 1990 when Expert Systems were seen as a solution for almost any problem, given a suitable Knowledge Base. Its downfall was caused by the systems inability to scale well to the addition of new set of rules and the incompleteness of such information in Knowledge Base [10].

Deep Learning excels in recognizing hidden patterns and became the modern go-to solution for complex problems such as image recognition [11], classification tasks [5], [11], [12] or estimations of human pose or age [13]. The drawbacks in this case are that it is susceptible to adversarial attacks [14] and presents the "black box of AI" problem due to the high number of auto-adaptive parameters and hyper-parameters. This approach was criticized for the lack of transparency on both the way the model produces results, and the logic over data obtained through training [15].

The proposed solution is a mixture of those two approaches. To overcome the deficiencies, Logic Tensor Network (LTN) [16] is a neuro-symbolic system that integrates symbolic knowledge and reasoning to deep learning models. This neuro-symbolic model supports learning, deductive reasoning and complex query answering, with a coverage of AI tasks similar to a neural network. This is possible by using Real Logic, a differentiable first-order logic language that interprets the semantics of LTN as Tensorflow computational graphs [17].

A. The LTN model

LTN semantics consists of groundings such as constants, variables, predicates, functions, connectives and quantifiers.

Constants in Logic Tensor Networks are depicted as tensors that can be of any rank. Tensors can be represented as n-dimensional arrays as shown below. Let the following:

$$\bigcup_{i_1,i_2,\ldots,i_k\in\mathbb{N}^*} \mathbb{R}^{i_1\times i_2\times\ldots\times i_k}$$

denote the set of tensors of any rank. The tensor of rank 0 represents a scalar value, a rank 1 tensor represents a one-dimensional vector, a rank 2 one represents a matrix, and so forth.

For the constant α let us refer to the associated tensor by $\mathbb{G}(\alpha)$. As an example, consider the constants x, y, z and the associated tensors $\mathbb{G}(x), \mathbb{G}(y), \mathbb{G}(z)$ of different ranks.

$$\mathbb{G}(x) = \begin{pmatrix} 1 & 2 \end{pmatrix}, \mathbb{G}(y) = \begin{pmatrix} 1 & 2 \\ 6 & 7 \end{pmatrix}, \mathbb{G}(z) = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Predicates in LTN are mapped to functions or tensor operations return a value in [0,1] that denotes a satisfaction level and can be interpreted as a truth degree. A predicate can be a simple λ function or a neural network of any shape and activation function (the only exception would be on the last layer, where a sigmoid function should be used to return a value of partial truth). Satisfiablity in LTN can be calculated by using an operator to aggregate the set of proposed formulas. The value of satisfiablity can be improved by modifying the groundings of such proposed formulas, also known as the process of learning. Connectives implemented in LTN are modeled to the semantics of first-order fuzzy logic [18]. Some of these are negation, implication, conjunction, and disjunction. Quantifiers supported in LTN are $\forall$(universal) and $\exists$ (existential) and are defined as aggregation functions. In the following experiments, only the $\forall$ p-mean error aggregator is used in training, and it is particularly helpful in aggregating the truth values of multiple formulas. A parameter p used in the function

$$X(a_1, a_2, a_3, ..., a_{k-1}, a_k) = 1 - \left(\frac{1}{k} \sum_{i=1}^{k} (1 - a_i)^p \right)^{\frac{1}{p}}$$

adjusts the strictness of the aggregator and a high value finds its use in querying where the focus is on the formulas, contrary to training where a lower value is used to prevent overtiffing and make it possible to generalise. For example, $p = 1$ would output the mean value, a $p = 2$ would result in an aggregation function that outputs the standard deviation of the inputs and a $p = \infty$ would output the minimum value.

Variables are defined as a sequence of individuals.

B. *Related work*

Some of the papers that do similar work in recognizing the potential of hybrid systems are [25], [27]. While in the latter one, the term ”hybrid system” is not directly used there, the authors explain how it is crucial for deep learning models to become able to make connections and associations in raw data, and to process them in a way suitable for further use. Also, they present how achieving that is possible by adding the benefits of Symbolic AI to Deep Learning models. Both papers also show the advancements of neuro-symbolic AI.

Another related work providing a comparison study between multiple algorithms is [26]. The authors also include an explanation of how the algorithms work on a specific dataset (in their case MNIST), then a part of experiments and in the end, they compare various results.

II. LTN SYSTEM FOR INTRUSIONS DETECTION IN NETWORK TRAFFIC

We designed and we experimented with a novel solution for characterizing IT security alerts based on LTN. By operating our model, we provide a detailed explanation of the results and we highlight the aspects that should be taken into consideration by anyone interested in the hybrid approach. We saw the potential of hybrid systems and used them in real-world scenarios such as the KDD99 dataset [21], to observe and interpret the results.

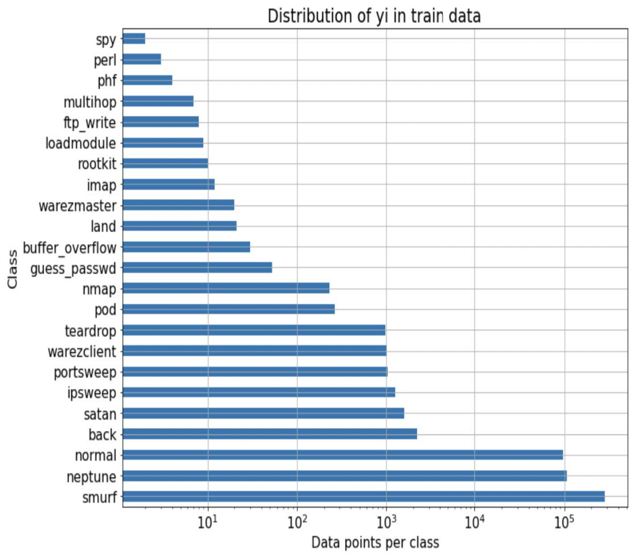

Fig. 1. Distribution of connection types before removal of duplicates

On the KDD99 dataset, our system seeks to discriminate between regular connections and intrusion attacks. KDD99 has 41 features and 494.020 entries, and it contains a wide range of network traffic connections. The ideal situation for an intrusion detection model is to identify the precise type of attack, however as shown in Figure 1, data for some attacks is insufficient, justifying the use of a logarithmic scale. It's worth noting that each link in our dataset contains an attribute named $label$ that shows the type of connection. We know that all attack types fit into the categories of DOS, R2L, U2R, or probe with the use of a Python Data Analysis Library called Pandas [20] and some prior information supplied in [22], [23].

As such, to help with the scarcity of some attacks and make a good prediction possible, we grouped all the intrusions by the category that they belong to. With **DOS (denial of service)** attack types, a huge amount of repeated server requests is sent with the malicious intent of using the victim resources, blocking any possible connections, or even forcing a shutdown/reboot. ”back”, ”land”, ”neptune”, ”pod”, ”smurf” and ”teardrop” are attacks from KDD99 dataset that fall into this category. **R2L** are attacks aimed to gather local data from another computer or server. The way this is done may be different, but the goal is the same. Attacks can try to query the computer/server, ”guess” the password via social engineering, or create local Trojan files to log any activity. In our dataset, ”ftp_write”, ”imap”, ”multihop”, ”phf”, ”spy”, ”warezclient” and ”warezmaster” are R2L attacks. **U2R** attacks happen when the attacker has access to another computer/user from the targeted network. It is meant to gain control of the root. This is possible by exploiting weaknesses in software and while it can be avoided, a complex software is prone to fall victim to this attack. In KDD99, ”buffer_overflow”, ”loadmodule”, ”perl” and ”rootkit” are attacks that target the root. **Probe** attacks are different from the ones mentioned before. Some of these attacks, such as ”ipsweep”, ”nmap”, ”portsweep” and ”satan” search for any possible weakness in the targeted computer. Usually in the form of connected IPs or open ports.

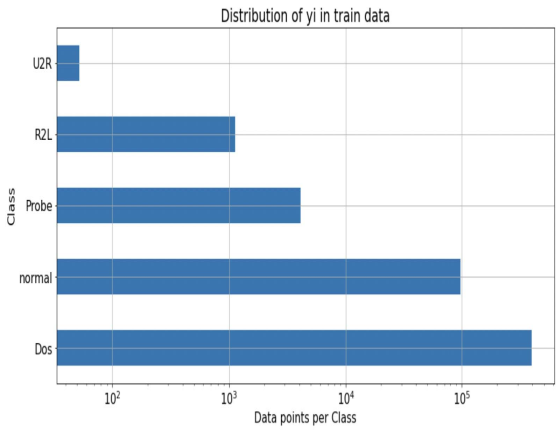

Fig. 2. Before removal of duplicates

A very helpful analysis of the KDD99 dataset and the impact of the features in intrusion detection can be found in [19]. Taking a look at that analysis and our exploration of the dataset, connections normal, smurf and neptune represent the vast majority and after grouping them, DOS and normal types are predominant as in Figure 2.

It is to be noted that most DOS attacks work by sending a large volume of traffic to the network system and while this attack category is without a doubt the most popular one (with smurf attack type being by far the most predominant one in this category) in the original dataset, after removal of duplicates (with the intent of reducing the size of the dataset and improving its quality) that is no longer true. The results after the removal of duplicates can be seen in Figure 3.

As Figure 3 shows, the distribution of data into classes is still far from being balanced. That being the case, during our experiments, we acknowledged that and tried using SMOTE oversampling technique [28] to increase the number of minority connection types [24]. While it did solve the scarcity issue, especially for U2R attacks, the computation power needed also greatly increased and resulted in a longer training time needed for the model. This tradeoff, at least in this specific example, did not seem worth it, as accuracy values were already good.

III. Experimental results

We normalized the data, one-hot encoded the categorical values, and maintained just the relevant features to appropriately categorize the attacks. The logic element of our neuro-symbolic framework is shaped by this feature significance, which allows us to apply Real Logic to identify potential threats. The relevant features are utilized to divide the dataset

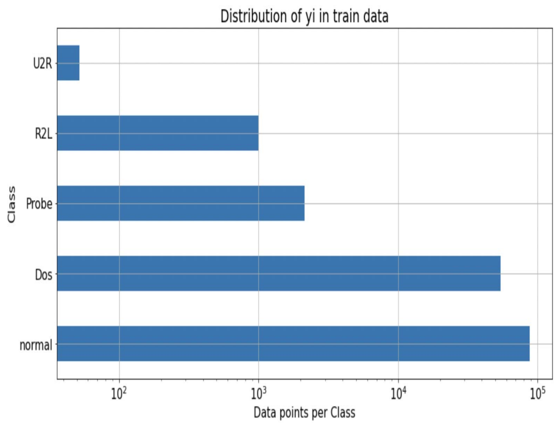

Fig. 3. After removal of duplicates

into a training set and a testing set, which will be used to train the model later on. As previously said, the purpose of this example is to demonstrate how to utilize LTN to discern between a regular connection and an attack, as well as categorize the latter.

This necessitates the definition of logic and the training of the model. As indicated previously, a predicate in LTN can be a neural network. In our case, a Deep Neural Network was used, but with the last layer having a ”softmax” activation function for the 5 newly grouped connection types. Normally, the aim of training is to maximize the loss value of that model, but for our neuro-symbolic model, the training objective is to maximize the satisfaction value of the Knowledge Base, composed by the set of axioms. The axioms are defined by each category of connections being represented by a label, denoting the class labels. We utilized the following set of Real Logic axioms to cover all potential cases and add additional constraints to make overfitting less likely.

$$\begin{gathered}
\forall x_{normal} : P(x_{normal}, normal) \\
\forall x_{DOS} : P(x_{DOS}, DOS) \\
\forall x_{probe} : P(x_{probe}, probe) \\
\forall x_{R2L} : P(x_{R2L}, R2L) \\
\forall x_{U2R} : P(x_{U2R}, U2R) \\
\forall x : \neg(P(x, normal) \wedge P(x, DOS)) \\
\forall x : \neg(P(x, normal) \wedge P(x, probe)) \\
\forall x : \neg(P(x, normal) \wedge P(x, R2L)) \\
\forall x : \neg(P(x, normal) \wedge P(x, U2R)) \\
\forall x : \neg(P(x, DOS) \wedge P(x, probe)) \\
\forall x : \neg(P(x, DOS) \wedge P(x, R2L)) \\
\forall x : \neg(P(x, DOS) \wedge P(x, U2R)) \\
\forall x : \neg(P(x, R2L) \wedge P(x, probe)) \\
\forall x : \neg(P(x, R2L) \wedge P(x, U2R)) \\
\forall x : \neg(P(x, probe) \wedge P(x, U2R))
\end{gathered}$$

An explanation for the proposed set of axioms is needed. In top down order:

- all the non-attacks should have label normal;
- all the DOS attacks should have label DOS;

- all the probe attacks should have label probe;
- all the R2L should have label R2L;
- all the U2R attacks should have label U2R;
- if an example x is labelled as normal, it cannot be labelled as DOS too;
- if an example x is labelled as normal, it cannot be labelled as probe too;
- if an example x is labelled as normal, it cannot be labelled as R2L too;
- and so forth...

Based only on the defined logic, we have calculated the initial satisfiability level to be 0.59471. We concluded training the model to detect intrusions is crucial. To make a comparison possible, most of the metrics used in training are similar with the exception of the 3 ϕ formulas.

1) The level of satisfiability of the Knowledge Base of the training data.
2) The level of satisfiability of the Knowledge Base of the test data.
3) The training accuracy (calculated as the fraction of the labels that are correctly predicted).
4) The test accuracy (same thing, but for the test samples).
5) The level of satisfiability of a formula we expect to be true. $\forall x : p(x, normal) \rightarrow \neg p(x, DOS)$ (every $normal$ connection cannot be a DOS connection and vice-versa)
6) The level of satisfiability of a formula we expect to be false. $\forall x : p(x, normal) \rightarrow p(x, probe)$ (every $normal$ connection is also a $probe$ one)
7) The level of satisfiability of a formula we expect to be false. $\forall x : p(x, normal) \rightarrow p(x, normal)$ (every $smurf$ connection has a $normal$ connection status)

The training metrics are stored in a *.csv* file that is also represented in Figure 4 and they can be interpreted as follow. In the first epoch, the level of satisfiability of the Knowledge Base increased from the initial value of 0.54725 to 0.6759 for the training data and 0.7216 for the test data. The results in the beginning of the training may seem atypical, with a higher accuracy for the test data, but it can be explained by the non-uniform intrusion distribution.

Satisfiability level in a neuro-symbolic models is similar to a loss function from deep learning and its evolution can be conceptualized as optimizing the model under relaxed first-order logical constraints. As the level of satisfiability for our Knowledge Base increased, the accuracy also improved, with values ranging from 0.7757 for our train data and 0.9162 for our test data in the first epoch and up to 0.9954 and 0.9947 respectively in the last epoch of training. Comparably, the accuracy for the standard deep neural network was 0.98863 for train data and 0.9869 for test data.

The set of Real Logic axioms that characterize the KDD99 dataset, was subsequently adapted also for the novel CIC-IDS2017 intrusion detection evaluation dataset [29].

For the neuro-symbolic model and the knowledge base in the CIC-IDS2017 scenario, we defined only that all the DDoS attacks should have a ”DDoS” label and all BENIGN connections should have a ”BENIGN” one and PortScan should have a ”PortScan” one. There was no need for constraints since the number of classes was small. The function grounded in LTN semantics was kept the same (a neural network with no change in the shape of hidden layers) and the metrics measured were also kept, with the removal of ϕ queries.

IV. Deep Neural Networks versus Logic Tensor Networks

In this section, we present in a compact way the advantages and disadvantages of both Deep Neural Networks and Logic Tensor Networks, for designing and constructing such a security alerts classification system.

A. Deep Neural Networks

Advantages of using a Deep Neural Network

- With little to no changes needed, it can be used on different applications and data types.
- Numerous articles [1]–[3] from trusted sources show amazing results in prediction, recognition and classifying.
- Depending on how the neural network is used, it can be supervised, unsupervised, semi-supervised or self-supervised. This covers most classification challenges.

Disadvantages of using Deep Neural Networks

- Lack of transparency on the model results. It is hard to understand how the input turned into the output. Black box problem represents one primary reason behind the shown interest in hybrid systems.
- To develop even a basic deep neural network, you need to have a good understanding of statistics, probability, linear algebra, calculus and data analytics.
- There is a high chance of causing overfitting. The model may focus on irrelevant details instead of the true underlying pattern that you wish to learn from the data. This leads to the trained model not being able to generalize well and have poor results in unseen data.

B. Logic Tensor Network

Advantages of using Logic Tensor Networks

- Logic Tensor Networks are able to solve most of neural networks tasks. Implementations of binary classification, single-label classification, multi-label classification, regression, clustering and prediction problems can be found on author’s repository [17].
- Hybrid systems require a reduced amount of training data. This is due to logic being defined beforehand that makes the neuro-symbolic framework to have a better initial state. Relational knowledge about objects, logical inference and ontological knowledge can be defined by Real Logic and make learning more efficient.
- Probably the most important aspect of Logic Tensor Networks is that Symbolic reasoning makes it possible for humans to understand how the model gave a specific output. One can query the satisfaction level of any defined formula and the results, query the constraints while also measuring the accuracy. The satisfaction level is depicted

with a fuzzy logic with values in [0,1], as opposed to the usual boolean value. The network uses the maximization of the satisfaction rules as an objective in learning by optimizing the groundings. Figure 4 shows it is possible

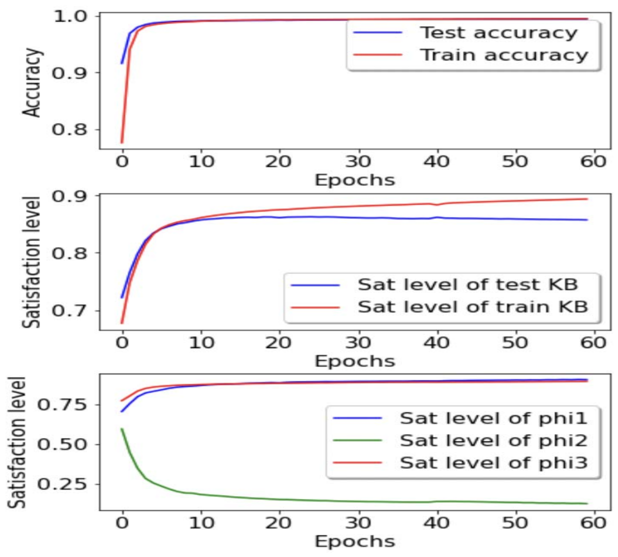

Fig. 4. Comparison of Accuracy, Satisfiability and Querying of Constraints for the LTN model

to have an accuracy metric (number of correct predictions divided by total number of predictions), specific to deep learning, but also have a satisfaction level of both the provided formulas and the queried constraints. Spikes in the first two graphs also make sense, as an incorrect prediction leads to a drop in the truth values of the axioms. While these are related, the accuracy level is almost perfect, while the satisfaction level can see some improvements. An ideal case to maximize both accuracy and satisfaction level of the Knowledge Base can only exist when the set of defined formulas perfectly resembles the correlation between the features, but that can prove to be difficult. Alt ought, this problem is mitigated by the ability of Deep Learning models to see the connections of features in data. LTN can also generalize and use learned knowledge when querying constraints and give impressive results.

- Overfitting can be avoided by providing accurate logic constraints in axioms.

Disadvantages of using Logic Tensor Networks

- While LTN can solve for example a multiclass classification it may not be the best alternative. KDD99, the dataset used in this paper has 42 types of attacks that need to be classified. It is entirely possible to relate them to each other but doing so may be cumbersome.
- As some problems may need extensive logic defined, it is to be noted that the number of axioms affects computational power.

Table 1 provides a side to side comparison of the Pros and Cons of Deep Neural Network versus Logic Tensor Networks.

Deep Neural Network	
PROS	**CONS**
flexible implementations for upcoming challenges	black box
impressive proven results	requires deep understanding of machine learning
can solve most classification challenges	overfitting can be a problem

Logic Tensor Network	
PROS	**CONS**
coverage of classification tasks	some problems require extensive logic defined
needs less data to train	scalability issue more axioms results in more time needed for training
interactive accuracy, learning and deductive reasoning possible	-
easier to avoid overfitting	-

TABLE I

V. Conclusions

The Logic Tensor Network proved to be able to give superior results for KDD99 dataset, with a very good precision in distinguishing between attacks and normal connections. It also adds another dimension in the interpretability of the model and the results, with Real Logic making it easier to understand how decisions were made. The experiments show that the addition of Symbolic Reasoning improves the quality of possible IDS based on hybrid systems and that advancements in the neuro-symbolic area are a tangible part of the future of Artificial Intelligence.

VI. Acknowledgements

The second author was supported by the project AISIMIA, contract no. 10163/11.06.2021, at the Politehnica University of Timisoara.

References

[1] Williams, CM, Rahul C, and Krishnan C., Cybersecurity risks in a pandemic, Journal of medical Internet research 22.9 : e23692, 2020.
[2] The Cost of Cyber Crime, https://purplesec.us/resources/cyber-security-statistics/#CyberCrime
[3] Ashoor, AS, and Sharad G., Importance of intrusion detection system (IDS), International Journal of Scientific and Engineering Research 2.1, pp 1-4, 2011.
[4] Haugeland, J., Artificial intelligence: The very idea, MIT press, 1989.
[5] LeCun, Y., Yoshua B., and Geoffrey H., Deep learning, Nature 521.7553, pp 436-444, 2015.
[6] Yan, R., Chen, X., Wang, P, and Onchis, D., Deep learning for fault diagnosis and prognosis in manufacturing systems, COMPUTERS IN INDUSTRY 110, pp.1-2, 2019.
[7] Diaz-del-Rio, F; Real, P and Onchis, DM., A parallel Homological Spanning Forest framework for 2D topological image analysis, PATTERN RECOGNITION LETTERS 83 , pp.49-58, 2016.
[8] Onchis, DM and Gillich, GR. Stable and explainable deep learning damage prediction for prismatic cantilever steel beam, COMPUTERS IN INDUSTRY 125, 2021.
[9] Schmidhuber, J., Deep learning in neural networks: An overview, Neural networks 61, pp 85-117, 2015.
[10] Bell, Michael Z., Why expert systems fail, Journal of the Operational Research Society 36.7, pp 613-619, 1985.
[11] Fujiyoshi, H., Tsubasa H., and Takayoshi Y., Deep learning-based image recognition for autonomous driving, IATSS research 43.4 : 244-252, 2019.

[12] Istin C, Doboli A. , Pescaru D. and Ciocarlie H., Impact of coverage preservation techniques on prolonging the network lifetime in traffic surveillance applications, 4th International Conference on Intelligent Computer Communication and Processing, 2008.

[13] Sun M., Kohli P., and Shotton J., Conditional regression forests for human pose estimation, in CVPR, pp. 3394–3401, 2012.

[14] Akhtar, N., and Ajmal M., Threat of adversarial attacks on deep learning in computer vision: A survey, IEEE Access 6 : 14410-14430, 2018.

[15] Hussain, J., Deep Learning Black Box Problem, Master Thesis Research Project, 2019.

[16] Serafini, L., and Artur d'Avila G., Logic tensor networks: Deep learning and logical reasoning from data and knowledge, arXiv preprint arXiv:1606.04422, 2016.

[17] Badreddine, S, et al. Logic tensor networks, Artificial Intelligence 303: 103649, 2022.

[18] Hájek, P., Metamathematics of fuzzy logic, Vol. 4. Springer Science & Business Media, 2013.

[19] Kayacik Günes H., Nur Zincir-Heywood A., and Heywood I. M, Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets, Proceedings of the third annual conference on privacy, security and trust. Vol. 94, 2005.

[20] McKinney, W., Data structures for statistical computing in Python, Proceedings of the 9th Python in Science Conference. Vol. 445. No. 1. 2010.

[21] KDD99 Dataset, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[22] KDD Cup '99 dataset (Network Intrusion) considered harmful,https://www.kdnuggets.com/news/2007/n18/4i.html

[23] Onchis D., Observing damaged beams through their time–frequency extended signatures, Signal Processing, Volume 96, Part A: 16-20, 2021.

[24] Onchis D., Feichtinger H., Constructive reconstruction from irregular sampling in multiwindow spline-type spaces. General Proceedings of the 7th ISAAC Congress, London, 2010.

[25] Garnelo, M., and Murray S., Reconciling deep learning with symbolic artificial intelligence: representing objects and relations, Current Opinion in Behavioral Sciences 29 : 17-23, 2019.

[26] Palvanov, A., and Young IC., Comparisons of deep learning algorithms for MNIST in real-time environment, International Journal of Fuzzy Logic and Intelligent Systems 18.2, : 126-134, 2018.

[27] Artur d'Avila G., and Lamb LC., Neurosymbolic AI: the 3rd wave, arXiv preprint arXiv:2012.05876, 2020.

[28] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research. 2002;16:321–57.

[29] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization", 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, January 2018