Detection of Insider Threats through visualization in a widespread dataset using Artificial Intelligence.

Vasileios Koutsouvelis, MSc Diploma of the Department of Computer Security and Networks of the School of Applied Sciences of the Open University of Cyprus.

****

Abstract **- Insider threats constitute one of the most important risk factors for Information Systems and the assets of an Organization. This paper presents results produced by the Convolutional Neural Network (CNN) algorithm, implemented via the Google Tensorflow program, which was trained to identify potential threats from images produced by an available dataset. The aim of the network education was to learn the appropriate weight values for all its neurons which were working to make a classification between a normal and a malicious activity. In this way the examination of the images that were produced and with the help of Machine Learning, the question whether the activity of each user is classified as "malicious" or not was answered.**

1. Introduction

Artificial Intelligence [1], [2], has been impressively evolving in recent years [3], [4], [5]. A typical example of applications based on Artificial Intelligence technologies are Artificial Neural Networks [6], [7], [8]. These systems are based on the properties of biological neural networks, and in particular, in the effort of people to understand some of the most basic functions and potentials of the biological brain, such as the coding of numbers, words, entities, concepts [9]. Artificial Neural Networks can play an extremely important role in the detection, prevention and handling of internal threats [10] and risks of information systems.

Today's era is marked by the particularly high growth and sweeping use of information and communication technology. Computers nowadays are used practically in every human activity and in all kinds of operations: commercial, scientific, etc. At the same time, however, the risks and cases of deliberate or accidental destruction, tampering or unauthorized use of data and computer resources in general are increasing. The consequences of possible destructions,

tampering or misuse of data can mean not only significant damage and costs, but also risks for the protection of the human rights of individuals [11].

The concept of security of an information system is related to the capability of an organization to protect its information against any tampering, destruction and unauthorized use of its resources. It also relates to its capability to provide accurate and reliable information, which is available to authorized users whenever they are looking for it.

Threat means a potential cause of security breach, which can cause damage to the system or the Organization. The threats are divided into the following categories:

- natural threats (e.g. fire, flood, earthquake),

- threats of an artificial nature (e.g. power failure, system and network software failure, application software failure, server failure, network device failure) and

- human threats, which are distinctly deliberate (e.g. misused identity by internal / external users or service providers, unauthorized use of the application, unauthorized modification of data, filtering of communications or interference with communications, theft of material, deliberate damage-vandalism) and accidental (e.g. erroneous deletion of files, incorrect routing, hardware or software maintenance error, introduction of malware) threats [10].

"Internal threat" refers to a person who may have privileged access to classified, sensitive or proprietary data and uses this advantage to maliciously mishandle or remove information from an organization and transfer it to unauthorized external users. These user cases include personnel of the company who bypass the control procedures for access to classified data.

2. Related Work

The detection of malicious activity with the help of Artificial Intelligence has been a matter of concern to scholars, who have used various methodologies to approach that from time to time.

Breier and Branisova (2015) [12] propose a threat detection method based on data mining techniques for analyzing system log files (log analysis). Their approach uses the "Apache Hadoop" technique, which allows the processing of large volumes of data in a parallel way.

This method detects new types of violations without further human intervention, while the overall error reaches a value below 10%.

Legg, Buckley, Goldsmith and Creese (2015) [13] propose a corporate threat detection system called "Corporate Insider Threat Detection (CITD)", which is the result of an interdisciplinary research project that incorporates technical and behavioral activities for the evaluation of threats caused by individuals. In particular, the system recognizes the users and the role-based profiles, and measures how users deviate from their observed behaviors in order to estimate the potential threat that a set of user activities can cause.

In another study, Sanzgiri and Dasgupta present techniques that have been developed to detect internal threats. In particular, it was pointed out that Hu proposed the modeling of the processes and users' activities to identify their divergent behavior. He used access control methods, based on roles to create rules of divergent behavior for each role. Giordano, on the other hand, observed users' behavior and analyzed it in order to verify if this behavior is the expected one. Armstrong uses the triangle Technology, Process and People (TPP) to propose a technique that uses social media data to detect internal threats. Kandias improved the model in order to identify users who are prone to such attacks and therefore suggest additional monitoring for these users. Maybury used users' behaviors beyond the ordinary limits, for example sharp economic robustness, frequent travels abroad, as an indication of opposing trends in the model of contradictory behavior. Greitzer used this contradictory model to predict internal threats by using a combination of psychological signs and abnormalities on the system's use. Eldardiry mentioned that current detection techniques are too specialized and do not take into account that the data monitoring users' behavior are vague, resulting in false negative results. Finally, the researchers remark that one of the main reasons why it is still difficult to detect attacks by internal users, is the lack of sufficient real available data in order to build and test models and mechanisms for detecting internal threats[14].

Furthermore, Tuor, Kaplan and Hutchinson (2017) [15] refer to a system of profound knowledge for filtering log files' data and analyzing them. According to the writers, an internal threat behavior varies widely, so the researcher does not attempt to formulate clearly the pattern of a behavior which is a threat. Instead, new variants of Neural Networks (DNN) and Recurring Neural Networks (RNNs) are trained to recognize the activity that is typical for each user on a network. At the same time, these Neural Networks assess whether the behavior of the user is

normal or suspicious. The authors note that detecting malicious cases is particularly difficult because attackers often try to imitate a typical behavior of a normal user.

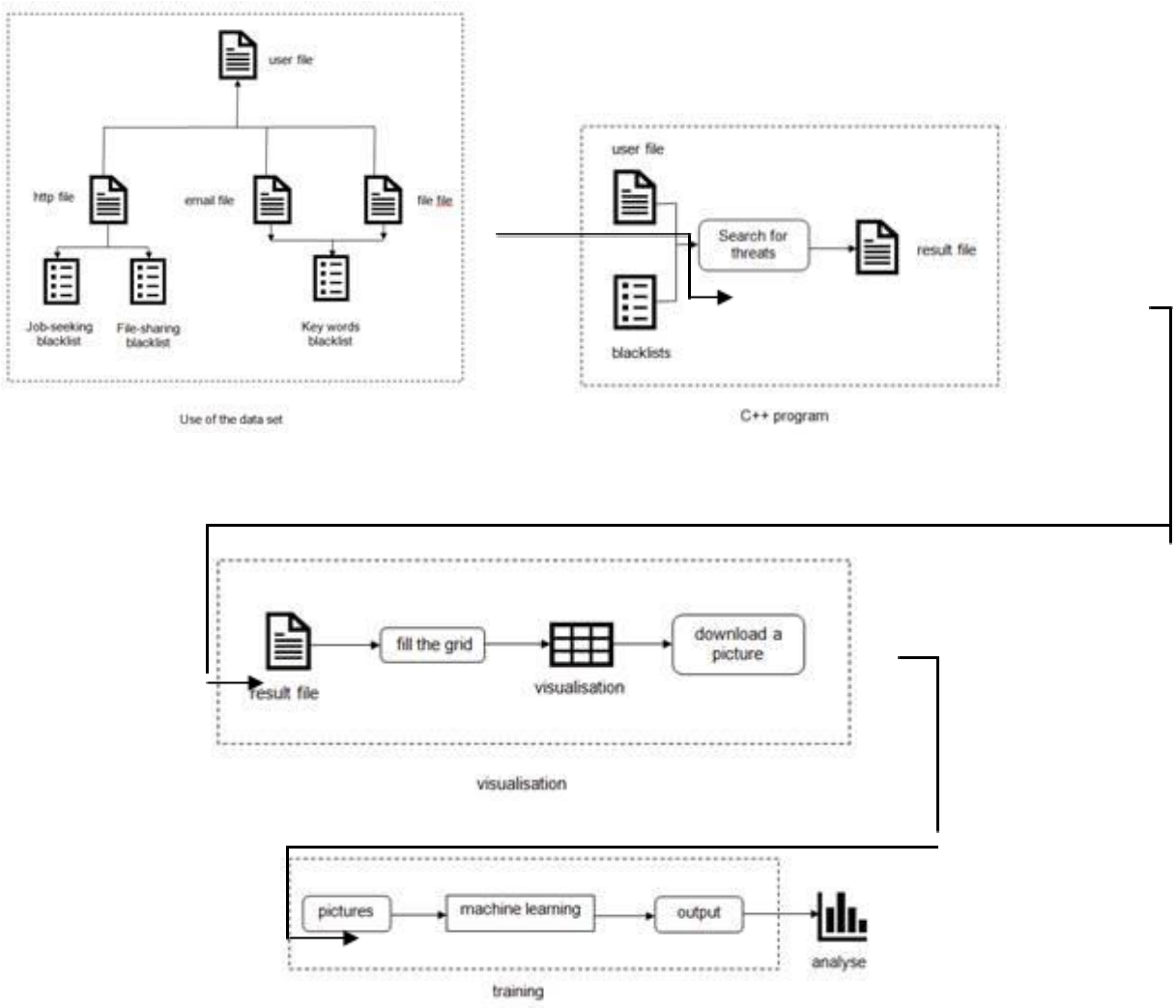
Some have used approaches [16] based on graphs to find malicious cases in structural data models that represent an internal threat activity, looking for activities that display similarities to normal data transactions but are structurally different from them. The work presented in [17] suggests approaches that combine Structural Anomaly Detection - SA. Contrary to the above methodologies, the work presented in this paper is based on the use of Machine Learning and training of the CNN algorithm to implement the categorization of user behavior into normal and malicious. Also, unlike other studies [18] focusing on domain knowledge to detect malicious behaviors through the use of specific SA, our methodology has focused exclusively on the behavior of each user of the Information Technology System. Apart from that, other methods [13] were based on collecting log data, by constructing a profile for each user and his / her role, while in our own, the user role did not have to be evaluated. Finally, the error rate in the method we used was very low (0,585) while other scholars [19] used methods where the error rate was greater.

3. Proposed Method

The methodology followed in this work detect internal threats in the available dataset, was to create images from the dataset, for a certain number of users. The images depicted the activity and behavior of each user as it emerged from the actions carried out in the Information System. A machine learning approach was then followed to classify a user's behavior as malicious or not. It should be noted that the characterization of a user's behavior as malicious is also dependent on the Security Policy that the Company or Organization adopts to protect its Information System. In this case, for example, we thought that visiting social networking sites or job search websites is a malicious activity. In addition, an important role in characterizing a user's activity is played by the position he/she holds in the Company / Organization. In the present case, as will be noted below, the activity of user PLJ1771, which - according to the dataset used - holds the position of IT Admin, has been classified as malicious activity, so that his/her position justifies - to some extent - the activity classified as malicious. Furthermore, the visualization of the data showed that the behavior of a user is not classified as malicious throughout the period of the investigation period. There were periods in which his/her activity complied with the rules governing the correct and lawful use of the Company's Information System for which he/she was working.

The steps we followed to complete the process and draw our conclusions were: 1) data sharing and creation of files based on the data of the user under consideration, 2) importing the data files we created in the D3.js library, selecting an appropriate image creation plan, examining the application library's patterns and creating images of the user in question that included his/her activity during each day, 3) creating images, 4) implementing and training the CNN algorithm in Tensorflow program and examining a user's behavior, which we have described as "normal" or "malicious"; and 5) drawing conclusions.

Figure 1 shows how this dataset is used in this study.



The dataset used was obtained from CERT [https://www.sei.cmu.edu/about/divisions/cert/index.cfm], which is part of the Institute of

Software Engineering (SEI). The file included log files, csv type, which recorded an activity covering a period of eighteen (18) months, archived on 01.01.2010 and expired on 31.05.2011. Through these files and after analysis and processing, it was attempted to present an image of the Information System and to analyze behaviors of users, which are described as malicious. The version that was received is 6.2r. The set of data consists of seven (7) parts - files, namely: 1) a file with the log in/logging of each user (logon.csv), 2) a file with the files used by the users (decoy_file. csv), 3) a file with the documents used by users (file.csv), 4) a file with the emails sent and received by the users (email.csv), 5) a file with http requests shipped by users (http.csv), 6) a file with the devices used by the users (device.csv) and finally 7) a file with the name of each user and his location in the Organization (psychometric.csv).

4. Experimental Results

In the first part of the experiment, a Java programming language application was created, which set out the search rules for the activity of the fifteen (15) users in question, who had the following names: 1) ACM2278 (Salesman), 2) CMP2946 (Salesman), 3) PLJ1771 (I.T Admin), 4) MBG3183 (Electrical Engineer), 5) SAB1954 (Mechanical Engineer), 6) ABK0481 (Software Quality Engineer), 7) CCB3055 (Administrator), 8) KCB1975 (Electrical Engineer), 9) ACG0312 (Production Line worker), 10) CJM0584 (Electrical Engineer), 11) SKB2635 (Computer Scientist), 12) CDE1846 (Electrical Engineer), 13) LAN2608 (Electrical Engineer), 14) POD0750 (Electrical Engineer) και 15) PIM3569 (Electrical Engineer). The application included four (4) steps that were linked to each other:

In particular, in step one, the program requested to import the file in which the user name that was to be searched (file.csv, email.csv, http.csv) was residing. The program searched every line of the file that was examined for the user's name that we had entered. If the user's name was found, it copied the line found and entered it into a new file. This file therefore included all the lines that the program had searched for and found. Eventually, three (3) files (file.csv, email.csv, http.csv) were created for each user, which included the total activity for the whole time period. In the second step, the results of the first step were entered into the program and the rules for the search for threats and malicious behavior were defined in each one of them. In the website category, we chose terms that were associated with social networks, work search sites, malware, and file sharing. In addition, in this category the program searched for attached files, which were divided into three (3) major categories: 1) files with size 50Kb - 100Kb, 2) files with size 100Kb

- 200Kb and 3) files larger than 200Kb. Finally, in the file category, we chose terms that were associated with malware (e.g. Keylogger), files that may have been leaked to the internet, and files that might have been distributed over the internet. The result was the creation of three (3) files per user (file.csv, email.csv, http.csv), which had the following structure: date, user (username), pc (from which the action was carried out), keyword (type of threat), number (the number corresponding to each threat) which files contained the respective threats. The number added at the end of each line of the above mentioned file defined a specific type of illegal activity. It corresponded to a unique color, which would be depicted in the next step in the visualization of the images that would be produced with the D3.js library in the user activity map. In Step Three we introduced the three (3) files per user, derived from the previous step. Our aim was to merge them into a single file that included all the activity of the individual user for the time period considered and which had the structure mentioned above. The fourth step was introduced into the program of the above file, which was produced in the third stage in order to separate the weekly and monthly activity per user. The files produced, which were of the same structure as above, included the user activity, both per week and per month for the time period considered in chronological order.

In the second stage of the experiment the final files of the previous stage were introduced to the Java D3.js library so as to visualize and produce images that depict the activity of each user in the form of a specific graph. As mentioned above, the number added at the end of each line of files created in the previous stage defined a specific type of activity, namely: 0: blue for job search, social networking sites, etc., 1: red for sharing site file-sharing websites, 3: pink for email threats, 4: green for file threats, 6: yellow for attachments 50Kb - 100Kb, 7 orange for attached files 100Kb - 200Kb and 8 : brown for attached files of size greater than 200Kb. Furthermore, Day / Hour Heatmap was chosen because it was the only one of the suggested types of imaging that included all days of the week and all hours of the day. This type was best suited to display week and month data. This experiment was based on two facts: On the one hand, were used specific scenarios - cases of threats that were contained in a file – which was part of the CERT files. According to these files, browsing work on web pages, uploading files to related sharing sites, and using malicious keylogger files were illegal activity. On the other hand, we considered as illegal activity, threats through emails as well as attachments of more than 200 Kb. At this point, of course, we would like to emphasize that each company defines a malicious activity in a different way which depends on its security policy.

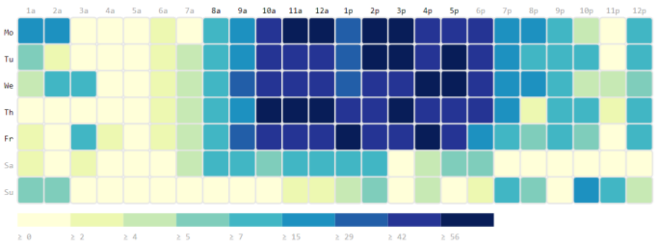


Figure 1: Type of the selected image

The aim of the program was the visualization of the files generated in the previous stage and then, the characterization of the activity, depicted either as malicious or as normal. The next step was to import the file data that came from the previous step. To implement this step, the d3.csv function of the D3.js. library was used. The function accepted csv files as the input of all users and carried out visualization based on the last field number of the csv file, which corresponded to a specific color and thus to a specific threat. In the last part of the program, the downloaded SVG element was created, converted to a png image and downloaded.

In the beginning, the image environment was designed and the dimensions required for proper image design were defined. In order to make things easier, the design was made in such a way that it includes the entire image so it does not need to be scrolled down with the mouse to display it. Furthermore, it was necessary to make the correct adjustment, based on the computer screen resolution, so that the image is clearly visible.

Then, the environment in which the imagery took place was designed. This included two (2) axes, of which the first (vertical) shows the days of the month and the second (horizontal) the hours of the day. The days, hours and colors used were stated at the beginning of the program code.

At the same time, the dimensions of the design area were determined according to the size of the grid that was set. The result was the creation of a space consisting of imaginary squares, each of which corresponded to each day and hour. Each time a square was activated, they were painted with the color which was set according to the number described, and the result was obtained.

The images that were produced were used in the final stage of the CNN algorithm training, through the TensorFlow program, for the automatic classification of activity into either normal or malicious. A total of eleven hundred and ninety-nine (1199) images were produced. We then

evaluated which of them were classified as malicious and which ones as normal. The selection was based on specific criteria, which were set by us, indicative of the density of the activity at specific time intervals, its colors, and the time recorded in the event of overtime. Finally, of all the above images, it was estimated that seven hundred and sixty-nine (769) of them contained malicious activity and four hundred and thirty (430) were classified as contacting normal activity. For indicative purposes, figures below present 6 such produced images, of which the three (3) first contain malicious activity and the three (3) following normal:

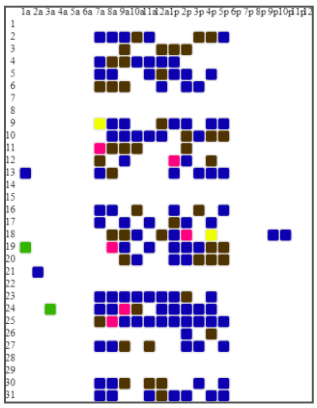


Figure 2: Monthly display of malicious user activity

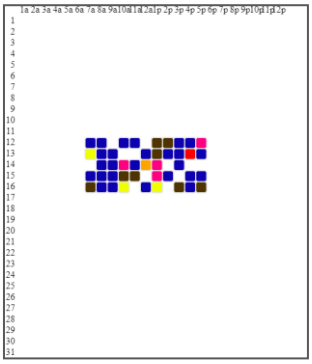


Figure 3: Weekly display of malicious user activity

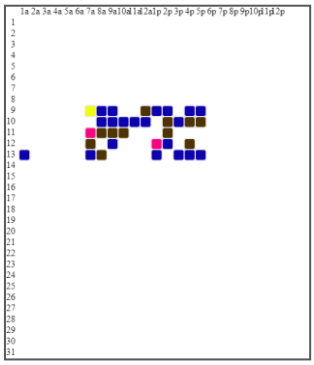


Figure 4: Weekly display of malicious user activity

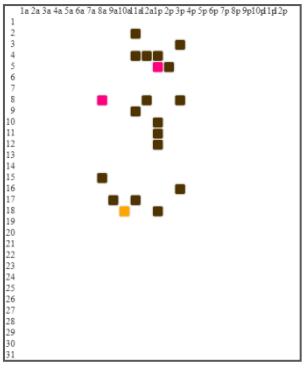


Figure 5: Monthly display of normal user activity

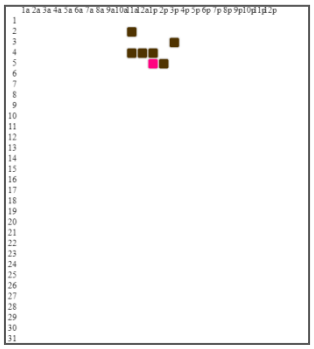


Figure 6: Weekly display of normal user activity

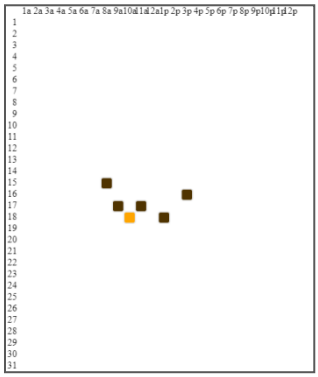


Figure 7: Weekly display of normal user activity

In the third stage of the experiment, Google's Tensorflow program was used to implement a CNN and image classification algorithm that occurred from the previous stage. The purpose was to implement a six (6) layer system that would recognize whether the image, which it would "read" as input, was classified as normal or malicious. The aim of the network education was to learn the appropriate weight values for all its neurons which were working to make a classification between a normal and a malicious activity. The system, which was implemented, is shown in the following figure:

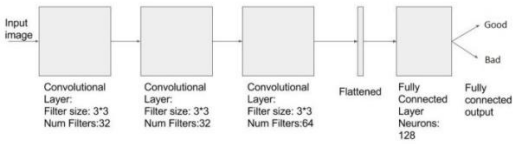


Figure 8: Schematic layout of the CNN algorithm implemented

Since, as mentioned above, one thousand nine hundred and ninety (1199) images were produced, of which four hundred and thirty (430) were estimated to contain physiological activity, consequently a corresponding number of malicious images was selected.

Finally, for the third stage of the experiment a total of eight hundred sixty (860) images were used, of which eight hundred and forty (840) were the ones for the training data of the algorithm as follows: 1) Training Data: 80% of the images were used for training; 2) Validation Data: 20% of the training images were used for validation; 3) Testing Data: The remaining twenty (20) images were the images for testing the Testing Data and reaching the conclusions of its forecasts.

672 images were selected, i.e. 80% for the training data of the algorithm and 168 images, i.e. 20% for the validation data. The reason why this percentage was chosen was due to the number of images to be used. The percentage that should be used for training data should be as large as possible so that the results produced are reliable and valid. Moreover, the percentage of validation data should be sufficient and not too low in order to give reliable results.

Then, one to twenty (20) images were introduced, ten of which contained - in our opinion - "normal" activity and a corresponding number of "malicious" activities. The forecasting of the results was absolutely successful, with a proportion that reached 100%.

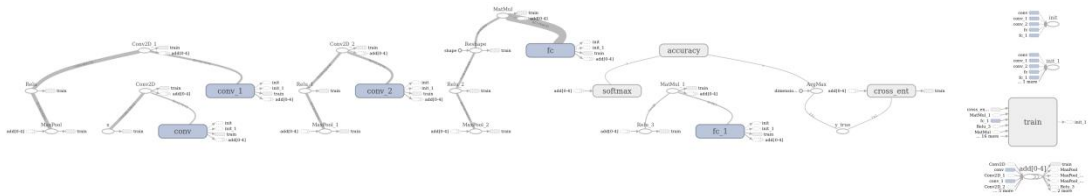


Figure 9: Block diagram of the implemented CNN algorithm

CNN Algorithm was traced graphically to illustrate training accuracy, validation accuracy, and cost. The illustrations were implemented with the Tensorboard program, a Tensorflow program package. Below are presented the three training exercises (training accuracy, validation accuracy, cost) and the implementation graph of the CNN algorithm:

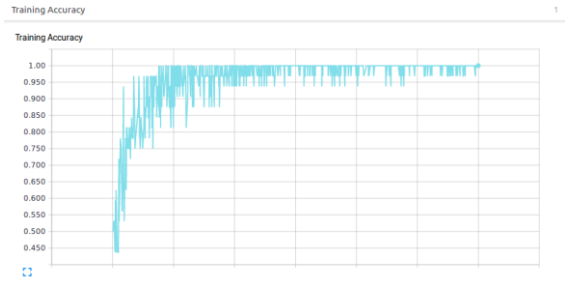


Figure 10: Training Accuracy

According to Figure 10, training accuracy starts at a value close to 0.450, ie 45%, and has an increasing value to reach a rate close to 100%.

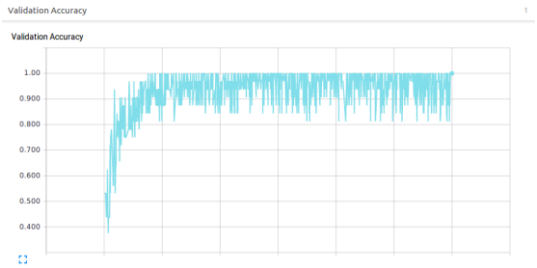


Figure 11: Validation Accuracy

Figure 11 highlights that validation accuracy starts from a value close to 0.400, ie 40%, and has a rising value to reach a rate close to 90%.

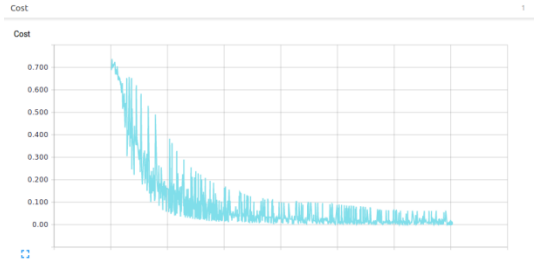


Figure 12: Cost

Figure 12 presents the cost that starts from a value close to 0.700, ie 70% and has a decreasing value to arrive at a value, which is close to 0.

In addition to the above percentage of the 80 - 20 % of managed images, we also tested percentages of 70 - 30 % and 60 - 40 %. The three (3) graphs of training accuracy, validation accuracy and cost, are shown below:

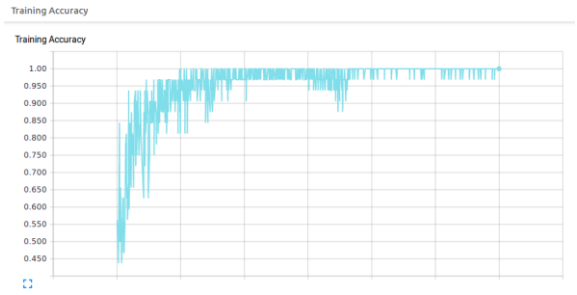


Figure 13: Training Accurracy

According to Figure 13, training accuracy starts at a value close to 0.450, ie 45%, and has an increasing value to reach a rate close to 100%.

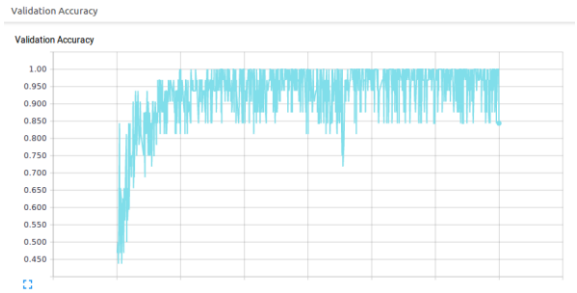


Figure 14: Validation Accurracy

Figure 14 highlights that validation accuracy starts from a value close to 0.45, i.e. 45%, and has a rising value to reach a rate close to 84 %.

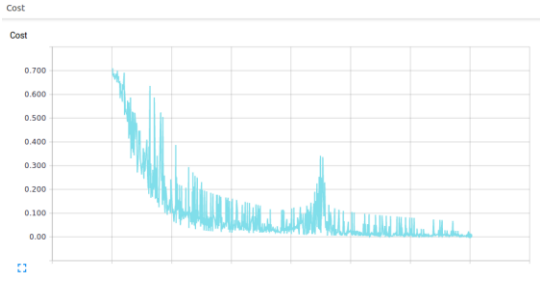


Figure 15: Cost

Figure 15 presents the cost that starts from a value close to 0.700, ie 70% and has a decreasing value to arrive at a value, which is close to 0.

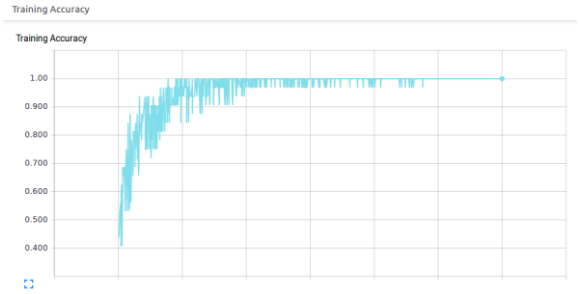


Figure 16: Training Accurracy

According to Figure 16, training accuracy starts at a value close to 0.400, i.e. 40 %, and has an increasing value to reach a rate close to 100%.

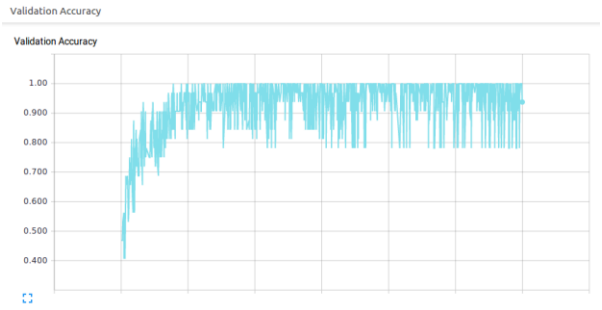


Figure 17: Validation Accurracy

Figure 17 highlights that validation accuracy starts from a value close to 0.400, i.e. 40%, and has a rising value to reach a rate close to 93%.

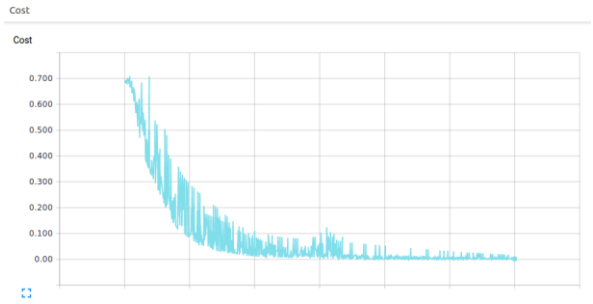


Figure 18: Cost

Figure 18 presents the cost that starts from a value close to 0.700, ie 70% and has a decreasing value to arrive at a value, which is close to 0.

Below is a concise table showing the results of the comparison of the percentages we used:

80 – 20 %	70 – 30 %	60 – 40 %
Testing Accuracy 100% Training Accuracy 100% Validation Accuracy 90.6% Cost 0.582	Testing Accuracy 100% Training Accuracy 100% Validation Accuracy 84% Cost 0.700	Testing Accuracy 100% Training Accuracy 100% Validation Accuracy 93% Cost 0.700

In addition to the different percentages, it is important to note that it took much more time to train the algorithm at the 70-30% and 60-40% share of available images, compared to 80-20%.

5. Conclusions

With this experiment, we managed a large amount of data of an information system, sorted them per user, visualized these data in order to create a system that will be able to examine the generated images and eventually classify them by characterizing activities as malicious or not.

Besides, in this experiment, a fairly large volume of actually available data was used to construct and test the model and its mechanism for detecting internal threats. Furthermore, these data covered a large period of one and a half years (01/2010 - 05/2011).

In addition, unlike other methodologies that used neural network variants and repetitive neural networks in order to recognize real-time user behavior, our model did not work in real time to recognize the malicious behavior at the time that occurs but it categorized a user's activity within the timeframe. In that way, we avoided the risk of mimicking the normal behavior by a malicious user in examining his behavior not only at a certain point in time but within a long period of time.

Additionally, unlike studies which are looking for activities that are similar to physiological data transactions but are actually structurally different from them and use graphs and theoretical approaches or statistical data processing to draw conclusions, our method used a practical

graphical approach, which has the advantage of having visualized users' activity per month and per week, thus making the modeling of malicious or inactive.

Other scholars also used the psychological PP profile as well as role-based profiles and suggested approaches that combined the detection of SA structural anomalies by social networks and information networks to observe and measure how users deviate from observed behaviors and if it can eventually be a potential threat. In our case, we only studied the activity of a user and his actions, without examining the psychological or other profile.

Finally, compared to other studies, in which data were examined in total, this experiment emphasized in the importance of data classification. For this reason, data sorting and categorization per user took place, in order to reduce the volume of data analysis, which contributed to better analysis and optimization of processing time.

Below is a concise table showing the results of some internal recognition methods, both in terms of validation accuracy and some of their comparisons with respect to this diploma dissertation:

Author/Paper	**Accuracy**	**Comparison**
Our Method	Testing Accuracy 100% Training Accuracy 100% Validation Accuracy 90.6% Cost 0.582	Mechanical learning (machine learning) and training of the CNN algorithm for the implementation and categorization of the user's behavior in normal and malicious.
W. Eberle, L. Holder, Detecting Insider Threats Using a Graph-Based Approach	Testing Accuracy 95%	In this method a graphical theoretical approach was used to detect malicious user behavior in an Information System
O. Brdiczka, J. Liu, B. Price, J. Shen, A. Patil, R. Chow, E. Bart, N. Ducheneaut, Proactive Insider Threat Detection through Graph Learning and Psychological Context	Server Eitrigg: 82.74% Server Cenarion Circle: 89.09% Server Bleeding Hollow: 79.84%	This study proposes an approach combining structural anomaly detection (SA) from social networks and information networks as well as the psychological profiling (PP) of individuals.
W. T. Young, H. G. Goldberg, A. Memory, J. Sartain T. Senator, Use of Domain Knowledge to Detect Insider Threats in Computer Activities	Indicators: 87.4% Anomalies: 97.9% Scenarios: 80.6%	This study focuses on the awareness of domain knowledge to detect malicious behaviors through the use of specific SAs algorithms.

J. Breier, J. Branisova, A Dynamic Rule Creation Based Anomaly Detection Method for Identifying Security Breaches in Log Records		The error rate of the method used was less than 10%. In the method we used, the error rate approaches zero (0).
Ph. A. Legg, O. Buckley, M. Goldsmith, S. Creese, Caught in the Act of an Insider Attack: Detection and Assessment of Insider Threat		The method used was based on collecting log data, by building a profile for each user and his / her property. In the method we used, the user property was not evaluated.

Table 1: Comparison of our method with others

References

[1] G. Krasadakis, Artificial Intelligence: What it is and how it dramatically changes our world, available in: http://www.naftemporiki.gr/story/1309187/texniti-noimosuni-ti-einai-kai-pos-allazei-dramatika-ton-kosmo-mas.

[2] J. Haugeland, Artificial Intelligence: Designing Mind: From Computational Theory to Modern Intelligent Engines, Publishing Katoptro, Athens, p. 38.

[3] M. Taddy, The Technological Elements of Artificial Intelligence, NBER Working Paper No. 24301, Issued in February 2018, available in: http://www.nber.org/papers/w24301.

[4] J. Borana, Applications of Artificial Intelligence & Associated Technologies, Proceeding of International Conference on Emerging Technologies in Engineering, Biomedical, Management and Science, ETEBMS – 2016, 5 – 6 March 2016, available in: https://pdfs.semanticscholar.org/d5b0/61e6565ce421b4b0b7d56296e882085dc308.pdf.

[5] R. A. Al, - Zaidy, C. L. Giles, A Machine Learning Approach for Semantic Structuring of Science Charts in Scholarly Documents, Proceedings of the Twenty - Ninth AAA I Conference of Innovative Applications (IAAI – 17), available in: http://www.aaai.org/ocs/index.php/IAAI/IAAI17/paper/download/14275/13716.

[6] M. Negnevitsky, Artificial Intelligence, Principles and Applications for the Development of Systems with Intelligent Technologies, Tziola Publications, Thessaloniki,, 2018[3], p. 166 - 167.

[7] I. Graham, Artificial Intelligence - at the cutting edge of science, Savvalas Publications, 2004, p. 9 - 12.

[8] A. Plerou, Artificial Neural Networks for Simulating the Human Brain, Open Education: the Journal for Open and Distance Education and Educational Technology, 2016, 8 (1), p. 128-135, available in: https://ejournals.epublishing.ekt.gr/index.php/openjournal/article/view/9794/9923.

[9] X. Schizas, K. Neocleous, Artificial Intelligence and Artificial Neural Networks, available in: http://www.pemptousia.gr/2017/08/texniti-noimosini-ke-texnita-nevronika-diktia/.

[10] S. Katsikas, Information Security Management, Pedio Publishing, Athens, 2014, p. 44 next.

[11] G. Pangalos - I. Mavridis, Security of Information Systems and Networks, ANIKOULA Publications, Thessaloniki, 2002, p. 16.

[12] J. Breier, J. Branisova, Anomaly Detection from Log Files Using Data Mining Techniques in: Lecture Notes in Electrical Engineering, January 2015, available in: https://www.researchgate.net/publication/282923954_Anomaly_Detection_from_Log_Files_Using_Data_Mining_Techniques.

[13] Ph. A. Legg, O. Buckley, M. Goldsmith, S. Creese, Caught in the Act of an Insider Attack: Detection and Assessment of Insider Threat in: IEEE International Symposium on Technologies for Homeland Security, Waltham, USA, 14th-16th April 2015, available in: https://www.researchgate.net/publication/276976957_Caught_in_the_Act_of_an_Insider_Attack_Detection_and_Assessment_of_Insider_Threat.

[14]. A. Sanzgiri, D. Dasgupta, Classification of Insider Threat Detection Techniques, available in: http://ais.cs.memphis.edu/files/papers/a25-Sanzgiri.pdf.

[15] A. Tuor, S. Kaplan, B. Hutchinson, Deep Learning for Unsupervised Insider Threat Detection in Structured Cybersecurity Data Streams, The AAAI-17 Workshop on Artificial Intelligence for Cyber Security WS-17-04.

[16] W. Eberle, L. Holder, Detecting Insider Threats Using a Graph-Based Approach, available in: https://pdfs.semanticscholar.org/482a/a61a5c417ce20a4bbf3e2a17d73f7f7afbb4.pdf.

[17] O. Brdiczka, J. Liu, B. Price, J. Shen, A. Patil, R. Chow, E. Bart, N. Ducheneaut, Proactive Insider Threat Detection through Graph Learning and Psychological Context, available in: https://www.parc.com/content/attachments/proactive-insider-threat-detection.pdf.

[18] W. T. Young, H. G. Goldberg, A. Memory, J. Sartain T. Senator, Use of Domain Knowledge to Detect Insider Threats in Computer Activities, available in: http://www.ieee-security.org/TC/SPW2013/papers/data/5017a060.pdf

[19] J. Breier, J. Branisova, A Dynamic Rule Creation Based Anomaly Detection Method for Identifying Security Breaches in Log Records, available in: https://www.researchgate.net/publication/284244364_A_Dynamic_Rule_Creation_Based_Anomaly_Detection_Method_for_Identifying_Security_Breaches_in_Log_Records.