Network Intrusion Detection System Performance Measurement Model

Timur Jamgharyan
National Polytechnic University
of Armenia
Yerevan, Armenia
e-mail: t.jamgharyan@yandex.ru

Abstract—**The paper presents the results of research performance test software for capsule, convolutional and generative-adversarial networks as part of network intrusion detection system. The research was conducted on datasets generated from *athena, dyre, engrat, grum, mimikatz, surtr* malware source code base. As a mathematical apparatus for reducing the dimension of calculated performance test metrics, the method of minimizing functions by the method of indefinite coefficients was chosen. The simulation of the developed software at different iterations and visualization of the results was carried out.**

Keywords—**Performance space, performance metrics, machine learning, dataset, malware, network intrusion detection system, metasploit.**

I. Introduction

When developing and deploying a network infrastructure (NI), an important place is occupied by the performance assessment of both the NI itself and its components. The performance of the entire NI is the sum of the performance of its individual components and their correct configuration. The network intrusion detection system (NIDS) as a component of NI, is a complex software and hardware complex, and evaluating its performance in different modes is also an important task. The NIDS workload is determined not only by the number of configuration rules (threat responses), but also by the degree of neutralization of these threats. In NIDS Snort 3.0 version, there are about 220 initial rules, based on which signature detectors of known threats and vulnerabilities from the database are activated [1]. Also, for NIDS, when evaluating performance, it is necessary to take into account the type of traffic being processed (network stack) and the bandwidth of the interfaces on which traffic is inspected. For hybrid NIDS or NIDS completely built on the basis of machine learning (ML), due to the probabilistic nature of the work of these NIDS, it is possible to name the exact value of the required value of the hardware resource only conditionally, with one or another probability. This problem is being actively researched at this stage [2-6]. In particular, in [2] a set of tests is presented, how fast systems can process input data and produce results using a trained model on different hardware platforms. A reference sample has also been developed to allow representative testing of a wide range of inference platforms and use cases. But this work reflects an assessment of the performance of neural networks in the processing of sound, language, pictures, text and machine vision. It should be noted that the theoretical assessment of the algorithmic complexity of the developed software algorithm does not always take into account the characteristics of the computer system. In all cases, the source code of the software is the starting point of the performance research. A qualitative performance test for neural networks is not just one experiment or one digit, it is a distribution of numbers.

The novelty of the researches lies in the performance test of capsule, convolutional, and generative adversarial networks based on malware datasets with a synthetically reduced dataset dimension. As a mathematical apparatus for reducing the dimension of calculated performance metrics, the method of minimizing functions by the method of indefinite coefficients was chosen.

The application of this method is due to the following factors:

- the method allows to reduce the dimension of the research data without loss of representativeness,
- the method allows binary calculation, which minimizes errors of the first and second kind (true positive, false positive, true negative, false negative) of the measured metrics.

II. Terms and Definition

A. Basic concepts

Definition 1: performance model - a model that includes all factors important for performance: source code, environment, input data and performance distribution [7].

Definition 2: «observer effect» - the influence of the observation process on the result [7].

Definition 3: Performance space in an object of study with a large number of dimensions, depending on many variables.

Definition 4: Profiling - collection of characteristics of the program.

B. Basic concepts of neural networks

Definition 1: Generative adversarial network (GAN) is an algorithm based on a combination of two neural networks, one of which generates an object, and the other tries to distinguish correct («real») objects from incorrect ones.

- the generating network *G* (generator) creates (generates) objects of a specified structure,
- the discriminating network *D* (discriminator) draws conclusions about the similarity of the generated and true objects [8-10].

The concept of generative adversarial networks, was invented in 2014 by Ian Goodfellow.

Definition 2: A convolutional neural network (CNN) is a specialized type of artificial neural network that uses a mathematical operation called convolution instead of the usual matrix multiplication in at least one of its layers. The structure of the network is unidirectional (without feedback), fundamentally multi-layered [11].

Definition 3: Capsule neural network (CapsNet) is a type of neural network that models hierarchical relationships [12].

III. Description of the Problem

Let there be a set of measurable performance metrics:

$$\left(\gamma_0^{(0,1)},\ \gamma_1^{(0,1)},\ \gamma_2^{(0,1)} \ldots\ldots\ldots \gamma_i^{(j)}\right) \in \Upsilon \qquad (1)$$

where,

$\gamma_i^{(j)}$ ($j = \{0,1\}$ -binary number) – performance metrics,

Υ - multidimensional performance space.

There is a native performance metric measurement function

$$\xi\left(\gamma_i^{(j)}\right) \in \Xi(\Upsilon) \qquad (2)$$

where,

$\xi\left(\gamma_i^{(j)}\right)$ - the output function of the measured performance of a single neural network,

$\Xi(\Upsilon)$ - the output function of measuring the performance of the entire «performance space»,

γ_1 metrics 1- request per second (RPS),

γ_2 metrics 2 - latency (time interval between the start and end of the NIDS triggering on malware),

γ_3 metrics 3 - throughput, (throughput of capsule, convolutional and generative-adversarial networks),

γ_4 metrics 4 - processor and random access memory (RAM) utilization. The measured set of metrics (f_γ) is minimized using the method of minimizing the output function by the method of undetermined coefficients. Let there be a dataset f_γ based on measured performance metrics $\gamma_i^{(j)}$.

It is necessary to reduce the dimension f_γ so that the function f'_γ formed on the basis f_γ is minimized without loss of representativeness (function f'_γ).

$$f_\gamma = \xi_1(\gamma_1, \gamma_2, \gamma_3, \gamma_4) = V(0,1,2,5,8,11,14,15)$$

$$f_\gamma = \xi_2(\gamma_4,\gamma_3,\gamma_2,\gamma_1) = \overline{\gamma_4\gamma_3\gamma_2\gamma_1} \vee \gamma_4\overline{\gamma_3\gamma_2\gamma_1} \vee \overline{\gamma_4}\gamma_3\overline{\gamma_2\gamma_1} \vee$$
$$\vee \overline{\gamma_4\gamma_3}\gamma_2\overline{\gamma_1} \vee \overline{\gamma_4\gamma_3\gamma_2}\gamma_1 \vee \gamma_4\gamma_2\overline{\gamma_2}\gamma_1 \vee \overline{\gamma_4}\gamma_2\gamma_2\gamma_1 \vee \gamma_4\gamma_3\gamma_2\gamma_1$$

$$\xi_0 = \gamma_1^0 + \gamma_2^0 + \gamma_3^0 + \gamma_4^0 + \gamma_{12}^{00} + \gamma_{13}^{00} + \gamma_{14}^{00} + \gamma_{23}^{00} +$$
$$+\gamma_{24}^{00} + \gamma_{34}^{00} + \gamma_{123}^{000} + \gamma_{124}^{000} + \gamma_{134}^{000} + \gamma_{234}^{000} + \gamma_{1234}^{0000} = 1$$

$$\xi_1 = \gamma_1^0 + \gamma_2^0 + \gamma_3^0 + \gamma_4^0 + \gamma_{12}^{00} + \gamma_{13}^{00} + \gamma_{14}^{01} + \gamma_{23}^{00} +$$
$$+\gamma_{24}^{01} + \gamma_{34}^{01} + \gamma_{123}^{000} + \gamma_{124}^{001} + \gamma_{134}^{001} + \gamma_{234}^{001} + \gamma_{1234}^{0001} = 1$$

$$\xi_2 = \gamma_1^0 + \gamma_2^0 + \gamma_3^1 + \gamma_4^0 + \gamma_{12}^{00} + \gamma_{13}^{01} + \gamma_{14}^{00} + \gamma_{23}^{01} +$$
$$+\gamma_{24}^{00} + \gamma_{34}^{10} + \gamma_{123}^{001} + \gamma_{124}^{000} + \gamma_{134}^{010} + \gamma_{234}^{010} + \gamma_{1234}^{0010} = 1$$

$$\xi_3 = \gamma_1^0 + \gamma_2^0 + \gamma_3^1 + \gamma_4^1 + \gamma_{12}^{00} + \gamma_{13}^{01} + \gamma_{14}^{01} + \gamma_{23}^{01} +$$
$$+\gamma_{24}^{01} + \gamma_{34}^{11} + \gamma_{123}^{001} + \gamma_{124}^{001} + \gamma_{134}^{011} + \gamma_{234}^{011} + \gamma_{1234}^{0011} = 0$$

$$\xi_4 = \gamma_1^0 + \gamma_2^1 + \gamma_3^0 + \gamma_4^0 + \gamma_{12}^{01} + \gamma_{13}^{00} + \gamma_{14}^{00} + \gamma_{23}^{10} +$$
$$+\gamma_{24}^{10} + \gamma_{34}^{00} + \gamma_{123}^{010} + \gamma_{124}^{010} + \gamma_{134}^{000} + \gamma_{234}^{100} + \gamma_{1234}^{0100} = 0$$

$$\xi_5 = \gamma_1^0 + \gamma_2^1 + \gamma_3^0 + \gamma_4^1 + \gamma_{12}^{01} + \gamma_{13}^{00} + \gamma_{14}^{01} + \gamma_{23}^{10} +$$
$$+\gamma_{24}^{11} + \gamma_{34}^{01} + \gamma_{123}^{010} + \gamma_{124}^{011} + \gamma_{134}^{001} + \gamma_{234}^{101} + \gamma_{1234}^{0101} = 1$$

$$\xi_6 = \gamma_1^0 + \gamma_2^1 + \gamma_3^1 + \gamma_4^0 + \gamma_{12}^{01} + \gamma_{13}^{01} + \gamma_{14}^{00} + \gamma_{23}^{11} +$$
$$+\gamma_{24}^{10} + \gamma_{34}^{10} + \gamma_{123}^{011} + \gamma_{124}^{010} + \gamma_{134}^{010} + \gamma_{234}^{110} + \gamma_{1234}^{0110} = 0$$

$$\xi_7 = \gamma_1^0 + \gamma_2^1 + \gamma_3^1 + \gamma_4^1 + \gamma_{12}^{01} + \gamma_{13}^{01} + \gamma_{14}^{01} + \gamma_{23}^{11} +$$
$$+\gamma_{24}^{11} + \gamma_{34}^{11} + \gamma_{123}^{011} + \gamma_{124}^{011} + \gamma_{134}^{011} + \gamma_{234}^{111} + \gamma_{1234}^{0111} = 0$$

$$\xi_8 = \gamma_1^8 + \gamma_2^0 + \gamma_3^0 + \gamma_4^0 + \gamma_{12}^{10} + \gamma_{13}^{10} + \gamma_{14}^{10} + \gamma_{23}^{00} +$$
$$+\gamma_{24}^{00} + \gamma_{34}^{00} + \gamma_{123}^{100} + \gamma_{124}^{100} + \gamma_{134}^{100} + \gamma_{234}^{000} + \gamma_{1234}^{1000} = 1$$

$$\xi_9 = \gamma_1^1 + \gamma_2^0 + \gamma_3^0 + \gamma_4^1 + \gamma_{12}^{10} + \gamma_{13}^{10} + \gamma_{14}^{11} + \gamma_{23}^{00} +$$
$$+\gamma_{24}^{01} + \gamma_{34}^{01} + \gamma_{123}^{100} + \gamma_{124}^{101} + \gamma_{134}^{101} + \gamma_{234}^{001} + \gamma_{1234}^{1001} = 0$$

$$\xi_{10} = \gamma_1^1 + \gamma_2^0 + \gamma_3^1 + \gamma_4^0 + \gamma_{12}^{10} + \gamma_{13}^{11} + \gamma_{14}^{10} + \gamma_{23}^{01} +$$
$$+\gamma_{24}^{00} + \gamma_{34}^{10} + \gamma_{123}^{101} + \gamma_{124}^{100} + \gamma_{134}^{110} + \gamma_{234}^{010} + \gamma_{1234}^{1010} = 0$$

$$\xi_{11} = \gamma_1^1 + \gamma_2^0 + \gamma_3^1 + \gamma_4^0 + \gamma_{12}^{10} + \gamma_{13}^{11} + \gamma_{14}^{10} + \gamma_{23}^{01} +$$
$$+\gamma_{24}^{01} + \gamma_{34}^{11} + \gamma_{123}^{101} + \gamma_{124}^{101} + \gamma_{134}^{111} + \gamma_{234}^{011} + \gamma_{1234}^{1011} = 1$$

$$\xi_{12} = \gamma_1^1 + \gamma_2^1 + \gamma_3^0 + \gamma_4^0 + \gamma_{12}^{11} + \gamma_{13}^{10} + \gamma_{14}^{10} + \gamma_{23}^{10} +$$
$$+\gamma_{24}^{00} + \gamma_{34}^{00} + \gamma_{123}^{110} + \gamma_{124}^{110} + \gamma_{134}^{100} + \gamma_{234}^{100} + \gamma_{1234}^{1100} = 0$$

$$\xi_{13} = \gamma_1^1 + \gamma_2^1 + \gamma_3^0 + \gamma_4^1 + \gamma_{12}^{11} + \gamma_{13}^{10} + \gamma_{14}^{11} + \gamma_{23}^{10} +$$
$$+\gamma_{24}^{11} + \gamma_{34}^{01} + \gamma_{123}^{110} + \gamma_{124}^{111} + \gamma_{134}^{101} + \gamma_{234}^{101} + \gamma_{1234}^{1101} = 0$$

$$\xi_{14} = \gamma_1^1 + \gamma_2^1 + \gamma_3^1 + \gamma_4^0 + \gamma_{12}^{11} + \gamma_{13}^{11} + \gamma_{14}^{10} + \gamma_{23}^{11} +$$
$$+\gamma_{24}^{10} + \gamma_{34}^{10} + \gamma_{123}^{111} + \gamma_{124}^{110} + \gamma_{134}^{110} + \gamma_{234}^{110} + \gamma_{1234}^{1110} = 1$$

$$\xi_{15} = \gamma_1^1 + \gamma_2^1 + \gamma_3^1 + \gamma_4^1 + \gamma_{12}^{11} + \gamma_{13}^{11} + \gamma_{14}^{11} + \gamma_{23}^{11} +$$
$$+\gamma_{24}^{11} + \gamma_{34}^{11} + \gamma_{123}^{111} + \gamma_{124}^{111} + \gamma_{134}^{111} + \gamma_{234}^{111} + \gamma_{1234}^{1111} = 1$$

Equating to zero all coefficients of zero rows, we remove them from the system.

$$\gamma_{123}^{000}+\gamma_{124}^{000}+\gamma_{234}^{000}+\gamma_{1234}^{0000}=1$$
$$\gamma_{123}^{000}+\gamma_{134}^{001}+\gamma_{1234}^{0001}=1$$
$$\gamma_{124}^{000}+\gamma_{1234}^{0010}=1$$
$$\gamma_{134}^{001}+\gamma_{1234}^{0101}=1$$
$$\gamma_{234}^{000}+\gamma_{1234}^{1000}=1$$
$$\gamma_{134}^{111}+\gamma_{1234}^{1011}=1$$
$$\gamma_{123}^{111}+\gamma_{1234}^{1110}=1$$
$$\gamma_{123}^{111}+\gamma_{134}^{111}+\gamma_{1234}^{1111}=1$$

In each equation containing minimal conjunctions, we equate the coefficients to zero.

$$\gamma_{123}^{111}\vee\gamma_{124}^{000}\vee\gamma_{134}^{001}\vee\gamma_{134}^{111}\vee\gamma_{234}^{000}=1$$

As a result of minimization, we obtain the following equation:

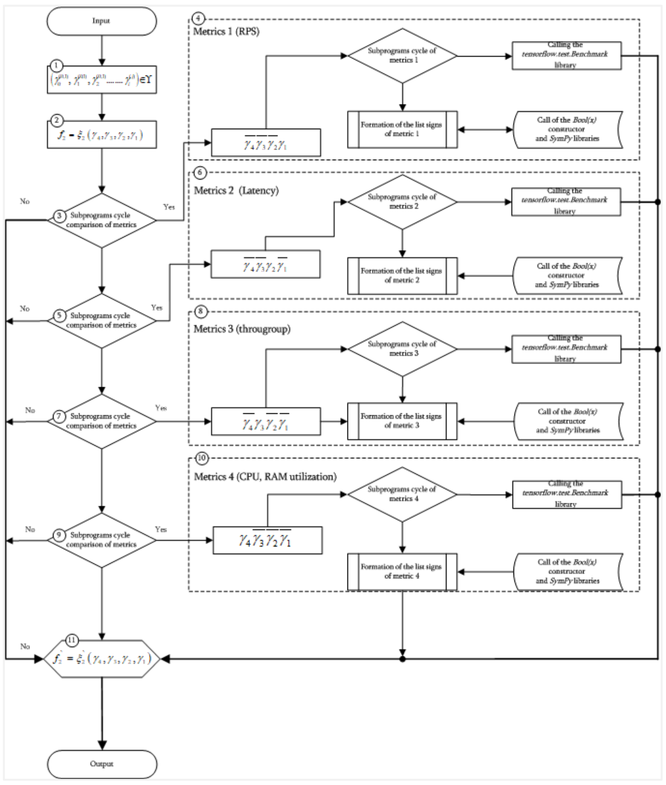
$$f_{\gamma}'=\grave{\xi}_2(\gamma_4,\gamma_3,\gamma_2,\gamma_1)=\overline{\gamma_4\gamma_3\gamma_1}\vee\overline{\gamma_3\gamma_2\gamma_1}\vee\overline{\gamma_4\gamma_2}\gamma_1\vee\gamma_4\gamma_2\gamma_1\vee\gamma_4\gamma_2\gamma_2$$

As a result of minimization, the number of measured sets of metrics is 2.13 times less than without minimization, which allows us to operate with a smaller sample size when conducting a performance test without reducing its representativeness.

$$f_{\gamma}=2{,}13f_{\gamma}' \qquad (3)$$

IV. Developed Algorithm

The developed algorithm is shown Fig. 1.



Input
Metrics 1 (RPS)
Metrics 2 (Latency)
Metrics 3 (througroup)
Metrics 4 (CPU, RAM utilization)
Subprograms cycle of metrics 1
Subprograms cycle of metrics 2
Subprograms cycle of metrics 3
Subprograms cycle of metrics 4
Calling the tensorflow.test.Benchmark library
Formation of the list signs of metric 1
Formation of the list signs of metric 2
Formation of the list signs of metric 3
Formation of the list signs of metric 4
Call of the Bool(x) constructor and SymPy libraries
Subprograms cycle comparison of metrics
Yes
No
Output

Fig. 1. The developed algorithm

Algorithm operation

At the input of the software that performruns, the performance test, datasets are preliminarily reduced to the JSON format (JSON - Java Script Object Notation).

Step 1: setting measurable performance metrics from the entire multidimensional performance space Υ,

Step 2: formation of parameters of measured performance metrics,

Step 3, 5, 7, 9: recursive loops of routines for comparing metrics. If minimization is possible, the corresponding event handler is activated (the value is «Yes») with a call to the *Sympy* library when the list of features for the measured metrics is formed. If it is impossible to minimize (the value is «No»), the output function is reconfigured (Step 11).

Step 4, 6, 8, 10: measuring the performance of given metrics and visualizing the results,

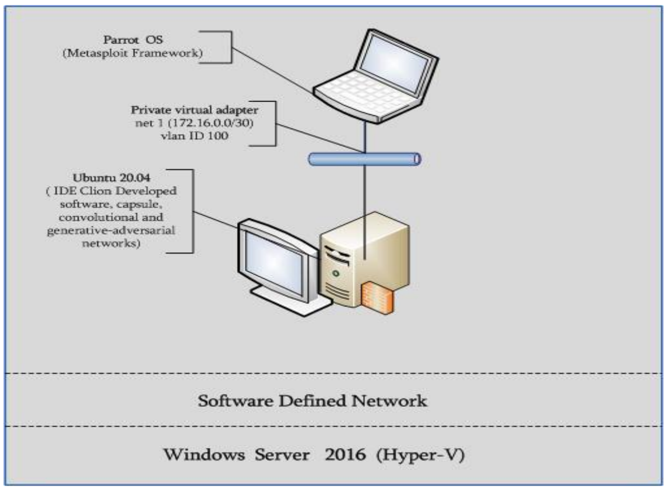
Step 11: generating an output function after performing a performance test based on the relevant metrics.

Performance test description

In a virtual environment based on the Windows Server 2016 Standart operating system (OS) [13], the Hyper-V role is installed in which the Parrot OS [14] OS with the Metasploit framework installed and the Ubuntu v20.04 OS in which the *Clion* development environment is installed: capsule, convolutional and generative-adversarial networks.

With the help of the Parrot OS obfuscated datasets malware *athena*, *dyre*, *engrat*, *grum*, *mimikatz*, *surtr* are gradually introduced. The NIDS is configured in discovery mode based on context triggered piecewise hashing. The introduction was carried out both individually and in complex data blocks of 20, 40, 80, 128, 256, 512, 1024 bytes [15]. The OS runs on a software-defined networking (SDN) in a Hyper-V virtualization environment and is connected to a Hyper-V virtual network adapter in *private* mode. Measurements for capsule, convolutional and generative-adversarial networks in malware detection mode at different values of context triggered piecewise hashing were carried out for 4 types of metrics ($\gamma_1\,\gamma_2\,\gamma_3\,\gamma_4$) at $\Upsilon=4$.

The scheme of measuring the performance test is shown in Fig. 2.



Parrot OS (Metasploit Framework)
Private virtual adapter net 1 (172.16.0.0/30) vlan ID 100
Ubuntu 20.04 (IDE Clion Developed software, capsule, convolutional and generative-adversarial networks)
Software Defined Network
Windows Server 2016 (Hyper-V)

Fig. 2. Scheme of measuring the performance test

V. Research Results

As a result of minimization, the characteristics of the NIDS are improved in terms of the parameters of metrics 1,2,3,4. As a result of minimization, there is an «exchange» of the degree of processor utilization for the amount of occupied RAM. *Athena*, *dyre*, *engrat*, *grum*, *mimikatz*, *surtr* were used as obfuscated malware [16-19]. The visualization of the results of ML NIDS performance tests is shown in Fig. 3 and Fig. 4. Detailed performance test output of the developed software using *TensorBoard* shown is Fig. 5.

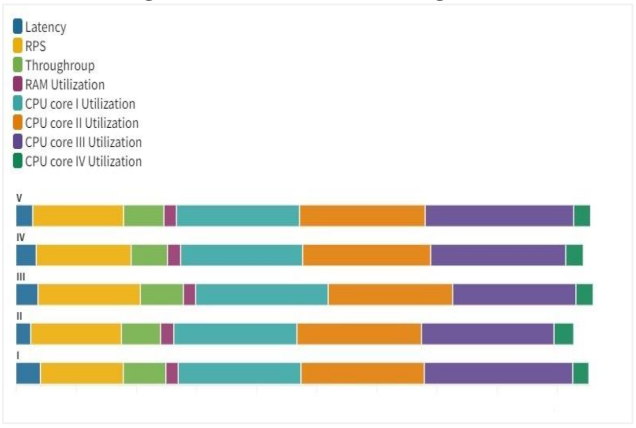


Fig. 3. Visualization of the ML NIDS performance tests without minimization

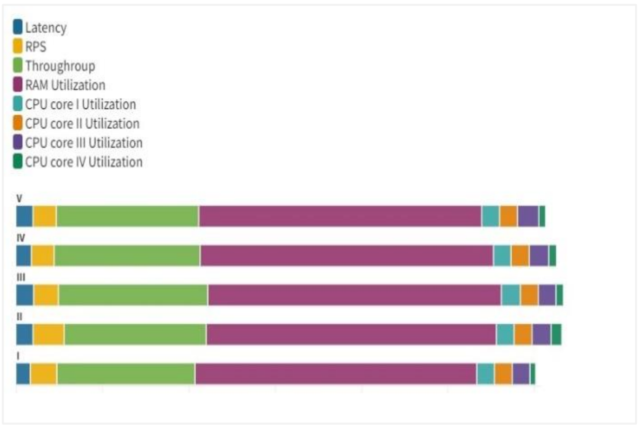
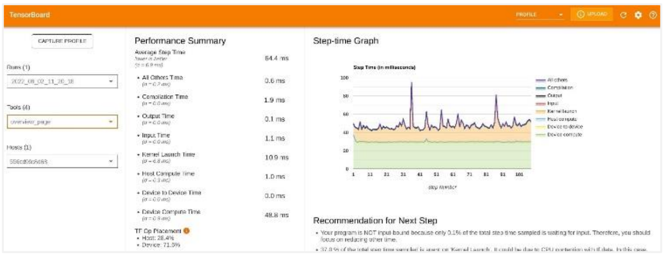


Fig. 4. Visualization of the ML NIDS performance tests using minimization



Performance Summary

```
Average Step Time          64.4 ms
lower is better
(σ = 6.9 ms)
 • All Others Time          0.6 ms
   (σ = 0.2 ms)
 • Compilation Time         1.9 ms
   (σ = 0.0 ms)
 • Output Time              0.1 ms
   (σ = 0.0 ms)
 • Input Time               1.1 ms
   (σ = 0.0 ms)
 • Kernel Launch Time      10.9 ms
   (σ = 4.8 ms)
 • Host Compute Time        1.0 ms
   (σ = 0.3 ms)
 • Device to Device Time    0.0 ms
   (σ = 0.0 ms)
 • Device Compute Time     49.8 ms
   (σ = 0.8 ms)
TF Op Placement
 • Host: 28.4%
 • Device: 71.6%
```

Recommendation for Next Step

- Your program is NOT input-bound because only 0.1% of the total step time sampled is waiting for input. Therefore, you should focus on reducing other time.
- 37.0 % of the total step time sampled is spent on 'Kernel Launch'. It could be due to CPU contention with tf.data. In this case,

Fig. 5. Detailed performance test output of the developed software using *TensorBoard*

VI. Conclusions

The paper considers a software model for reducing the dimensionality of datasets for conducting a software performance test using the method of minimizing the output functions of data sets by the method of undetermined coefficients. The developed software makes it possible to test the performance of capsule, convolutional and generative-adversarial networks within the specified metrics. The performance test of the most developed performance software was carried out on the basis of the *TensorFlow* open library tensorflow.test.Benchmark, visualization using the *TensorBoard* plugin.

Based on the tests carried out, we can conclude:

Minimizing the output functions of datasets by the method of undetermined coefficients is justified with a predetermined performance space and exactly known metrics. In all cases, there is an increase in the performance of capsule, convolutional and generative-adversarial networks within the given metrics (regardless of the number of iterations). An increase in performance entails an increase in the consumption of hardware resources, in particular RAM. With a decrease in the dimension of output data sets by an average of 2÷2.5 times, there is an increase in RAM consumption by 18%÷25%, but at the same time, the level of processor utilization decreases. Software source code and all research results are available at [20].

References

[1] Official website of intrusion detection and intrusion prevention system Snort. [Online]. Available: https://www.snort.org

[2] Vijay Janapa Reddi, Christine Cheng et al. MLPerf Inference Benchmark.[Online].Available: https://arxiv.org/abs/1911.02549

[3] V.Kustikova, E.Vasiliev et al. DLI: Deep Learning Inference Benchmark.[Online].Available: https://link.springer.com/chapter/10.1007/978-3-030-36592-9_44

[4] G.Cheng,H,Yuan et al: Towards Large-Scale Small Object Detection:Syrvey and Benchmarks [Online].Available: https://arxiv.org/abs/2207.14096

[5] Y.Shen,L.Wang et al: An Interpretability Evaluation Benchmark for Pre-trained Language Models [Online].Available: https://arxiv.org/abs/2207.13948

[6] S.Pontes-Filho et al: Towards the Neuroevolutional of Low-level Artifical General Intelligence. Online]. Available: https://arxiv.org/abs/2207.13583

[7] A.Akinshin, *Pro. NET Benchmarking, The Art of Performance Measurement*, first edition (2019) // 576, APRESS®

[8] Ian J.Goodfellow,J.Pouget-Abadie, M. Mirza, B.Xu, D.Warde-Farley,S.Ozair, A. Courville,Y.Bengio. *Generative Adversarial Networks*. [Online]. Available:https://arxiv.org/abs/1406.2661

[9] A. Gad. Fatima E. Jarmouni. *Learning and Neural Networks with Python™. A Practical Guide.* ACADEMIC PRESS. Elsevier, 2021

[10] R.T.Kneusel. *Practical Deep Learning. A python – based introduction.* San Francisco. 2021

[11] L.Stefan,V. Fjodor. (2020). The Neural Network Zoo. *Proceedings. 47. 9. 10.3390/proceedings47010009.*

[12] G.Hilton, S.Sabour, N,Frosst, «Matrix Caplsules with EM Routing»,(2018), https://research.google/pubs/pub46653/

[13] Microsoft official site. Windows Server 2016 Operating System Download Page [Online].Available: https://www.microsoft.com/en-us/evalcenter/download-windows-server-2016

[14] Parrot OS official site. Parrot OS Operating System Download Page [Online].Available: https://www.parrotsec.org/

[15] T. V. Jamgharyan, «Research of Obfuscated Malware with a Capsule Neural Network». *Mathematical Problems of Computer Science*, vol. *58*, pp.67–83. 2022.

[16] Malware database. [Online]. Available http://vxvault.net/ViriList.php

[17] Malware repository. [Online]. Available https://avcaesar.malware.lu/

[18] Viruses repository. [Online]. Available: https://virusshare.com/

[19] Malware repository. [Online]. Availablehttps://github.com/ytisf/theZoo

[20] Software source code and all research results. [Online].Available: https://github.com/T-JN?tab=repositories