Performance Analysis of Machine Learning Platforms Using Cloud Native Technology on Edge Devices

Konrad Cłapa[1][0000−0002−6939−6504], **Krzysztof Grudzień**[2][0000−0003−4472−8100], **Artur Sierszeń**[2][0000−0001−8466−4856]

[1]*Atos Poland R&D Sp. z o.o. Bydgoszcz, Poland*
konrad.clapa@atos.net
[2]*Lodz University of Technology, Institute of Applied Computer Science, Poland*
krzysztof.grudzien@p.lodz.pl, artur.sierszen@p.lodz.pl

DOI:10.34658/9788366741928.31

Abstract. *This article presents the results of an experiment performed on a machine learning edge computing platform composed of a virtualized environment with a K3s cluster and Kubeflow software. The study aimed to analyze the effectiveness of executing Kubeflow pipelines for simulated parallel executions. A benchmarking environment was developed for the experiment to allow system performance measurements based on parameters, including the number of pipelines and nodes. The results demonstrate the impact of the number of cluster nodes on computational time, revealing insights that could inform future decisions regarding increasing the effectiveness of running machine learning pipelines on edge devices.*

Keywords: *Machine learning, Artificial intelligence, Cloud computing, Edge computing, Internet of Things*

1. Introduction

Machine learning (ML) and artificial intelligence (AI) on edge computing devices are among the most dynamically developing research areas. The importance of the research results is directly related to the development of deep learning methods, the architecture of data processing hardware modules and the increase of large datasets availability. The scientific researches related to applied artificial intelligence for every aspect of daily life and a growing number of sampling and measuring IoT systems generate the possibility of creating many innovative solutions [1]. Those can improve the experience, quality of life, and development progress in many industry verticals, e.g. medical and manufacturing [2].

For the sovereignty-constrained solution, it is essential that the machine learning models can be created and managed regardless of the underlying platform (public cloud, data centre). Fig. 1 is presented a cloud-native solution, based on the containerisation of machine learning pipelines, that brings in an abstraction layer which allows running the machine learning model on hybrid and distributed platforms. To coordinate the execution of machine learning pipelines, the cloud-native platforms require so-called orchestrators that introduce overhead in resource consumption, including memory and processor usage [3, 4].

An increase in the effectiveness of running machine learning pipelines without extending the hardware capabilities of the edge device is critical for serving the machine learning models in real time. Therefore, it is essential to research and design the software elements of the solution that will minimise the overheads and reduce the delays caused by the orchestrator for the cases where edge devices are used to provide sovereignty for data processing and analysis. The state-of-the-art technologies enabling machine learning (ML) on edge systems are K3s, MicroK8s and Kubeflow [5].

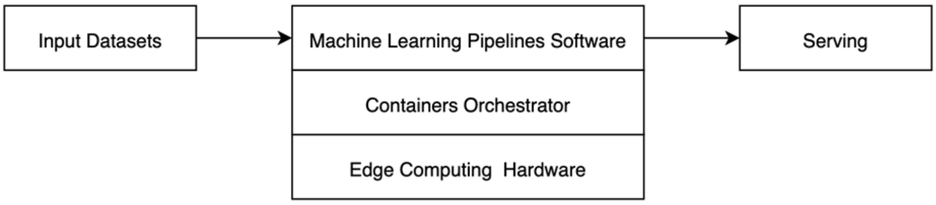

Figure 1: Machine Learning Pipelines on Edge Devices Architecture. Source: own work.

2. Scope of research

In the first phase of the research, the effectiveness of the existing software solutions for managing the lifecycle of the machine learning pipelines is being conducted. The long-term plan consists of several steps. The next phase of the research of the critical elements related to the system's effectiveness will be identified for the case of the platform that cannot use public cloud resources due to data protection restrictions. Finally, we will pinpoint the bottlenecks and perform research to improve the effectiveness of the elements that introduce overheads and delays. The result of the study will include (i) identification of methods of improving the effectiveness of running machine learning pipelines on edge devices allowing protection of confidential data, (ii) a model of architecture based on cloud-native technologies, (iii) a prototype for a particular industry use case (iv) documented proof for improvement of the effectiveness of running machine learning pipelines on edge devices.

3. Laboratory environment setup

The experimental laboratory setup consisted of two layers – hardware and software. Regarding hardware choices, a decision has been taken to build a lab environment that will characterise itself with maximised stability for testing the system. We virtualised all the Kubernetes cluster nodes and got the static configuration to the nodes based on the Ubuntu Linux distribution, the environment setup was defined in a code, and a rebuild of the virtualised environment could be executed significantly faster. In the next step, based on testing of stability (failure like suspension, unexpected switch off, etc.) K3s was chosen as a Kubernetes distribution.

The last stage of the laboratory environment study resulted in continuing experiments with Kubeflow Pipelines components of Kubeflow. This element is crucial to perform ML operations. Installing additional features of the Kubeflow package can only put overhead on the cluster and will not contribute to the experiment's candidate will perform. On each of the machines, there were K3s nodes installed. All nodes hosted control planes and actual workloads and constituted a fully operational Kubernetes cluster. Finally, the Kubeflow Pipelines were deployed on that cluster. The components of the Kubeflow pipelines were distributed to the node by the Kubernetes Scheduler, and no affinity or anti-affinity rules have been applied.

4. Experiment description

The main research was focused on identifying how the number of nodes in an edge Kubernetes cluster can affect the efficiency of executing the pipelines. We decided to run simple arithmetical computations within containers and measure the pipeline's time to execute. A varying number of pipelines was run parallel.

The experiment started with a single execution, ending with k=30 pipelines executed in parallel. These executions have been conducted on clusters with 1-3 nodes. This allowed analysing both how execution time increases when we increase the number of parallel executions and what is the effect of increasing the nodes number. Each execution was run at least ten times. For the experiment, a benchmarking script was developed. The script allowed us to run multiple requests to the Kubeflow API to create and execute computational pipelines. The time of the execution of a container was measured directly by the Kubeflow software, so any latency related to network connection or data generation could be avoided. The scheme of the experiment environment setup is presented in Figure 2. The hardware used for the experiment has not been used for any other computation at the time of the experiment to avoid interruptions. The number of nodes was controlled by powering up and down virtual machines on a Qemu hypervisor.

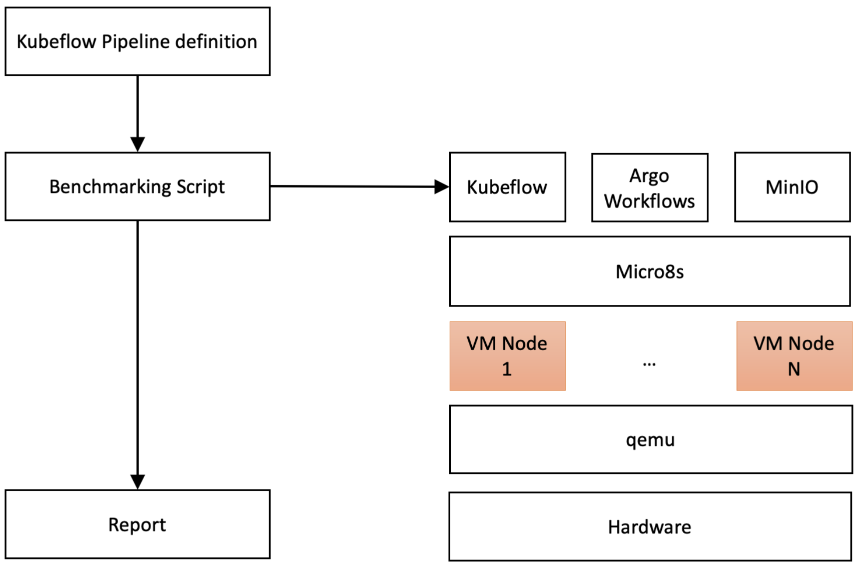

Figure 2: Experiment environment setup. Source: own work.

5. Experiment results

The experiment results of the exaction for N=1-3 nodes are presented in 3. The collected data are visualised on the plots, with the minimum, maximum and average of the executions values. The plotted results indicate that the execution times display almost linear growth with an increase in the number of parallel executions. The behaviour was expected as the load on the system increased. Additionally, we observed that increasing the number of nodes in the system had a very low impact on the execution time, with execution times being almost the same for up to 10 parallel runs. Additionally, one can observe minimal performance increase from 11 parallel runs when we compare 1 node system with 2 and 3 nodes. But it isn't essential. We also see that the difference between a 2 and 3 nodes system is virtually indistinguishable.

6. Conclusions

Based on conducted experiments, it can be concluded that the simple computational tasks in the pipeline are not generating enough load on the nodes to justify the increased effectiveness of the system with an increase in the number of nodes.

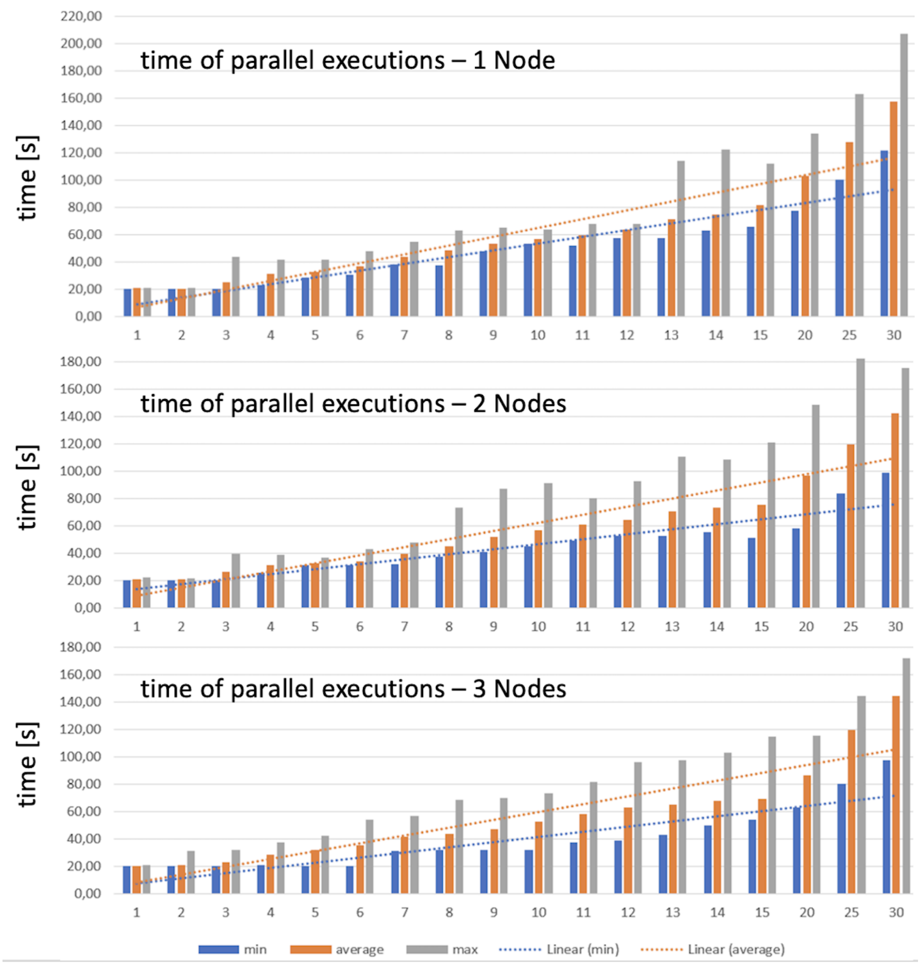

Figure 3: Experiment results for N node systems. Source: own work.

The analyses of results show that increasing the number of nodes in the system had a shallow impact on the execution time. However, such an effect was visible. It is necessary to expand the laboratory environment regarding increased nodes and prepare more demanding calculations. The future study will be focused on testing the system with more computational power-demanding workloads and measuring additional parameters like the pipeline creation time to reflect the system's performance indicators.

Acknowledgment

This research is conducted as a part of the 5th edition of the Implementation Doctorate program financed by the Polish Ministry of Education and Science.

References

[1] Xiong J., Chen H., *Challenges for building a cloud native scalable and trustable multi-tenant aiot platform*, [In:] *Proceedings of the 39th International Conference on Computer-Aided Design*, ICCAD '20, Association for Computing Machinery, New York, NY, USA, 2020, doi: 10.1145/3400302.3415756.

[2] Lv Z., Chen D., Lou R., Wang Q., *Intelligent edge computing based on machine learning for smart city*, *Future Generation Computer Systems*, 2021, vol. 115, pp. 90–99, doi: 10.1016/j.future.2020.08.037.

[3] Rausch T., Rashed A., Dustdar S., *Optimized container scheduling for data-intensive serverless edge computing*, *Future Generation Computer Systems*, 2021, vol. 114, pp. 259–271, doi: 10.1016/j.future.2020.07.017.

[4] Toka L., Dobreff G., Fodor B., Sonkoly B., *Machine learning-based scaling management for kubernetes edge clusters*, *IEEE Trans. on Netw. and Serv. Manag.*, 2021, vol. 18, no 1, p. 958–972, doi: 10.1109/TNSM.2021.3052837.

[5] Fathoni H., Yang C.T., Chang C.H., Huang C.Y., *Performance comparison of lightweight kubernetes in edge devices*, [In:] *Pervasive Systems, Algorithms and Networks*, Springer International Publishing, Cham, 2019, pp. 304–309.