SERVERLESS VS. ON-PREMISES: A PERFORMANCE ANALYSIS OF ML DEPLOYMENT WITH AWS FARGATE, GCP CLOUD RUN, AND ON-PREM

OĞUZ KIRÇİÇEK

MEF UNIVERSITY

APRIL 2024

MEF UNIVERSITY

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

MASTERS'S IN INFORMATION TECHNOLOGIES

M. Sc. THESIS

SERVERLESS VS. ON-PREMISES: A PERFORMANCE ANALYSIS OF ML DEPLOYMENT WITH AWS FARGATE, GCP CLOUD RUN, AND ON-PREM

Oğuz KIRÇİÇEK

ORCID NO: 0000-0001-5530-8625

Thesis Advisor: Asst. Prof. Dr. Tuna ÇAKAR

APRIL 2024

ACADEMIC HONESTY PLEDGE

I declare that all the information in this study is collected and presented in accordance with academic rules and ethical principles, and that all information and documents that are not original in the study are referenced in accordance with the citation standards, within the framework required by the rules and principles.

Name Surname: Oğuz KIRÇİÇEK

Signature:

ABSTRACT

SERVERLESS VS. ON-PREMISES: A PERFORMANCE ANALYSIS OF ML DEPLOYMENT WITH AWS FARGATE, GCP CLOUD RUN, AND ON-PREM

Oğuz KIRÇİÇEK

M.Sc. in Information Technologies

Thesis Advisor: Asst. Prof. Dr. Tuna ÇAKAR

April 2024, 61 Pages

In this study, I present a comparative analysis of the changes occurring during the deployment process of machine learning models, both in On-Premises systems and cloud service providers. The successful deployment of machine learning models holds critical importance for businesses and organizations aiming to enhance their productivity. Understanding and comparing how models behave in different environments is of paramount significance to make informed decisions.

Prominent commercial organizations like AWS and GCP offer reliable and cost-effective cloud services tailored to provide customized web applications. Our primary objective in this article is to guide cloud customers by highlighting the key features of the most recognized Cloud Service Providers and facilitating informed decision-making through comparisons with the On-Premises option. Additionally, I explore the advantages of managed services such as AWS Fargate and Google Cloud Run, which streamline application deployment.

Through this research, my goal is to offer useful insights that help companies succeed in the fast-paced, cutthroat business environment by helping them make wise strategic decisions.

Keywords: Cloud, On-Prem, AWS, GCP, Performance Analysis

Numeric Code of the Field: 92404

ÖZET

"SUNUCUSUZ VS YERİNDE: AWS FARGATE, GCP CLOUD RUN VE YEREL ORTAMDA MAKİNE ÖĞRENİMİ DAĞITIMININ PERFORMANS ANALİZİ"

Oğuz KIRÇİÇEK

Bilişim Teknolojileri Tezli Yüksek Lisans Programı

Tez Danışmanı: Dr. Öğr. Üyesi Tuna ÇAKAR

Nisan 2024, 61 Sayfa

Bu çalışmada, makine öğrenimi modellerinin dağıtım sürecinde meydana gelen değişikliklerin On-Premises sistemler ve bulut hizmet sağlayıcılarındaki karşılaştırmalı analizini sunmaktayım. Makine öğrenimi modellerinin başarılı bir şekilde dağıtılması, üretkenliklerini artırmayı amaçlayan işletmeler ve kuruluşlar için kritik bir öneme sahiptir. Modellerin farklı ortamlarda nasıl davrandığını anlamak ve karşılaştırmak, bilinçli kararlar vermek için büyük öneme sahiptir.

AWS ve GCP gibi önde gelen ticari organizasyonlar, özelleştirilmiş web uygulamaları sunmak üzere tasarlanmış güvenilir ve maliyet etkin bulut hizmetleri sunmaktadır. Bu makalenin temel amacı, en tanınmış bulut hizmeti sağlayıcılarının anahtar özelliklerini vurgulayarak bulut müşterilerini yönlendirmek ve On-Premises seçeneği ile karşılaştırmalar yaparak bilinçli karar almayı kolaylaştırmaktır. Ayrıca, AWS Fargate ve Google Cloud Run gibi yönetilen hizmetlerin avantajlarını keşfediyoruz, bu hizmetler uygulama dağıtımını kolaylaştırmaktadır.

Bu araştırma aracılığıyla, işletmelerin stratejik kararlar alarak dinamik ve rekabetçi iş dünyasında başarı elde etmelerine olanak tanıyan değerli içgörüler sağlamak amaçlanmıştır.

Anahtar Kelimeler: Cloud, On-Prem, AWS, GCP, Performans Analizi

Bilim Dalı Sayısal Kodu: 92404

Tüm süreç boyunca bana destek olan

aileme ve arkadaşlarıma

ithaf ediyorum.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

AI : Artificial Intelligence

IaaS : Infrastructure as a Service

SaaS : Software as a Service

PaaS : Platform as a Service

CaaS : Containers as a Service

FaaS : Function as a Service

IT : Information Technology

API : Application Programming Interface

IBM : International Business Machines Corporation

ML : Machine Learning

AWS : Amazon Web Services

GCP : Google Cloud Platform

ECS : Elastic Container Service

EKS : Elastic Kubernetes Service

CPU : Central Processing Unit

GKE : Google Kubernetes Engine

GB : Gigabyte

TB : Terabyte

SSD : Solid State Drive

MS : Millisecond

MB : Megabyte

HTTP : Hypertext Transfer Protocol

I/O : Input/Output

IAM : Identity and Access Management

Hyper-V : Hypervisor-based Virtualization

md5 : Message Digest Algorithm 5

VM : Virtual Machine

OS : Operating System

MS Azure : Microsoft Azure

SQL : Structured Query Language

DB : Database

SNS : Simple Notification Service

SQS : Simple Queue Service

S3 : Simple Storage Service

VPC : Virtual Private Cloud

DNS : Domain Name System

NOSQL : Not Only Structured Query Language

EC2 : Elastic Compute Cloud

RDS : Relational Database Service

EBS : Elastic Block Store

EMR : Elastic MapReduce

GUI : Graphical User Interface

ME : Millennium Edition

XP : Experience

NT : New Technology

R2 : Release 2

RAM : Random Access Memory

PC : Personal Computer

SDK : Software Development Kit

vCPU : Virtual Central Processing Unit

GPUs : Graphics Processing Units

GCF : Google Cloud Functions

VM : Virtual Machine

Max : Maximum

Avg : Average

ESU : Ephemeral Storage Utilization

GIB : Gibibyte

INTRODUCTION

In today's fast-changing world of technology, businesses and software developers really need cloud services to keep their applications running well, fast and reliably. This is extremely important because it helps them make their applications better and cheaper in a highly competitive environment. Using cloud services is important because it changes the way applications are built, makes things run more efficiently and reduces costs.

The latest technological advances in this area contribute to making cloud computing even more widespread and powerful.

Recent developments will see the integration of artificial intelligence and machine learning into cloud computing. In the future or now, AI can be used to automatically deploy, scale and optimize products on the cloud.

While in the early years of cloud computing we had to depend on a single cloud provider, in the future hybrid cloud systems or multi-cloud systems will and even have entered our lives. Hybrid cloud systems are an approach that allows companies to combine their own data centers with cloud computing. Currently, there are different cloud providers in the market and with the developing technology, companies will be able to combine and use them at the same time. In the past, companies were hesitant about security in cloud computing. It is predicted that there will be significant developments in security in the coming years and this problem will be solved. The use of cloud computing is increasing among developing countries such as Turkey, which means that in these countries, cloud computing helps businesses to digitalize and increase their competitiveness.

All these developments indicate that cloud computing will play an even more important role in the future. Cloud computing will help businesses operate more efficiently and effectively, adopt new technologies and expand into new markets.

When choosing what cloud providers, any business has to learn or know the pros and cons of cloud providers according to their own needs. First of all, companies and

developers should know the requirements of the system needed in the context of IaaS, SaaS & PaaS.

Infrastructure as a Service (IaaS) is a model that provides IT infrastructure resources like computing, storage, and network on a pay-as-you-go basis over the internet. This model allows businesses and users to rent infrastructure resources in a cloud environment. The management and maintenance of the infrastructure are taken care of by the IaaS provider, and users are not involved in this aspect. Users play a role in application development and deployment layers. This model offers great flexibility for quickly starting projects and meeting business requirements. Some of the IaaS providers include Google Cloud Platform, Amazon Web Services, and Microsoft Azure.

Software as a Service (SaaS) refers to services that are provided over the cloud, without the need to download any software. Fast internet is the only requirement for SaaS to work. Examples of such software include Outlook and Hotmail, and you only need internet access to use them. SaaS is suitable for small and individual businesses as it allows them to quickly start using the software instead of installing and managing it. This approach provides advantages to businesses in terms of cost, productivity, and application deployment. It operates on a pay-as-you-go principle, meaning you pay for what you use.

Platform as a Service (PaaS) should not be confused with SaaS. It's a set of cloud services used to build and manage application software. The most significant advantage of this model is that it allows users to access the software required for application development and operation without having to purchase, set up, and maintain the underlying infrastructure. There are other advantages as well, including lower costs, faster application development, multiple platform integration, API development and management, and more. Examples of PaaS include Amazon Web Services, IBM Cloud, and Google App Engine.

With the increasing volume of data, deriving meaningful insights from data has become crucial. The concept of machine learning has emerged to address these issues, assisting companies in developing sales strategies by extracting insights from data.

Processing big data through machine learning algorithms can yield valuable output, enabling companies to formulate strategies. Machine learning algorithms can be applied in various sectors, including recommendation engines, disease detection, and vehicle recognition systems, among others. However, for these models to continuously function, they need to be deployed and run in specific environments. At this point, different solutions have been offered. Comparing the impact of serverless and on-premises infrastructure on the deployment of machine learning models is of great importance. Companies need to find the most optimal way for their models, and this study follows a similar path by conducting research on the functioning of a machine learning model using serverless and on-premises services, with some metrics as references.

Serverless architecture has gained significant popularity in recent years, with cloud providers offering services supporting serverless architectures. This approach allows businesses to free themselves from infrastructure management and focus solely on running applications. Serverless architecture provides advantages such as automatic scaling and fault tolerance.

On the other hand, On-Premises systems represent a traditional approach where companies host machine learning models in their own data centers or physical servers. In this model, businesses have complete control and advantages in areas like security and data privacy. However, the On-Premises approach also entails managing and updating infrastructure, which may require additional resources and result in higher costs.

While exploring the flexibility, scalability, and cost-effectiveness of the Serverless architecture, I also examine the advantages of On-Premises in terms of control and security, evaluating how each approach differentiates in terms of performance and efficiency. This comparison aims to assist companies in making informed and strategic decisions while formulating deployment strategies in the professional world.

In this article, I present an analysis comparing the impact of Serverless and On-Premises infrastructures on the deployment of machine learning models that use Docker. Containerization provides a standardized and portable environment, allowing ML models

to be deployed and run consistently across various platforms. Hence, the ML model has been deployed on containers, enabling us to build and run Docker containers, and utilize Cloud Run and AWS Fargate on cloud platforms. On the On-Premises side, the ML model has been deployed on Windows using Docker. This article will provide comparisons of these three different systems, considering similar machine specifications, and conducting the comparison on 5-6 key aspects.

Cloud Run: Developed by Google Cloud, Cloud Run is a CaaS platform that allows serverless deployment and execution of containers. It supports various programming languages such as Go, Node.js, Python, Java, and .NET Core. Cloud Run automatically handles cluster creation and infrastructure management, eliminating the need for manual intervention.

AWS Fargate: Developed by Amazon Web Services, AWS Fargate is another CaaS platform that operates serverlessly. It is compatible with Elastic Kubernetes Service (EKS) and Amazon Elastic Container Service (ECS). AWS Fargate enables application deployment without server management.

Windows: In this project, the machine learning model runs on On-Premises systems that require maintenance and infrastructure management. Since this project targets similar cpu, memory and zone utilization, Windows is deployed on Google Compute Engine.

Some metrics were used for comparison between cloud providers and on prem systems such as Response time, memory usage, CPU usage, network and more. These metrics are important for showing the use of application, functionality, and performance. Memory usage and CPU utilization are key indicators for indicating the resource consumption of an application. Response times are a factor affecting the user experience and speed of service.

When choosing what cloud providers to use, any business must learn or know the pros and cons of cloud providers according to their own needs. The path followed in this

study will help developers and businesses to decide to a certain extent the cloud systems they will use.

The remainder of this study is structured as follows: I'll give an overview of the technologies in Section 1. The suggested container-based design is shown in full in Section 2, along with information on the cloud providers. The experimental progress and evaluation criteria are presented in Section 3. The benchmark results are described in Section 4. In Section 5, the subject is covered, and the current state of the art is assessed along with related works.

a) Purpose of Thesis

The purpose of this paper is to serve as a reference for studies conducted in two different cloud environments and an on-premises environment. In this study, you will observe how an ML model running on Docker is deployed in various environments and examine its performance. AWS Fargate, GCP Cloud Run, and Windows are utilized for comparison. The primary objective of this study is to conduct a performance measurement between Serverless systems and On-Premises, and some of distinct performance metrics have been established: response time, network, memory usage, CPU usage and more. Performance analysis has been carried out based on these metrics.

b) Literature Review

In recent years, the rapid growth of data has driven advancements in the field of machine learning, leading to increasingly complex models. Successfully deploying these models has become of critical importance. To address this challenge, developers have explored various deployment options. Serverless and On-Premises systems are significant choices in this context.

Krzysztof Burkat et al. [1] made comparisons on AWS Fargate, AWS Lambda and GCP Cloud Run services. In this study, the authors aimed at performance comparison between Fargate and Lambda, comparison of limitations between Cloud Run and Fargate and evaluation of Hybrid approaches and compared these 3 services in 4 different scientific workflows. According to the results, Lambda is clearly faster than Fargate. The reason for this is that Fargate can take up to 60 seconds to stand up the container. Fargate is not recommended for small jobs. The location for Lambda and Fargate was chosen EU-Ireland. Lambda is charged for every 100ms of execution. Fargate is charged per second. They concluded that Fargate is not the optimal solution for small-grained, low-memory tasks. They believe that Lambda can perform better. However, they caution that Fargate can be allocated more parameters such as memory and disk space. Fargate has a faster physical core than Cloud Run in the experiments. In the study, Cloud Run has 2048MB-2vCPU and Fargate has 4096MB-2vCPU. There were no significant problems or delays in Cloud Run execution. Average execution times are given. Studies have argued that Cloud Run is a better solution than Fargate, but they caution about the smaller resource pool and maximum execution time limit for Cloud Run. Fargate and Lambda have tried a hybrid approach together and got very good results. For Cloud Run, they report that it does not allow enough memory to handle large tasks. As a result, the authors observed that Fargate introduces a container setup overhead of 60 seconds and the API has a throttling limit that prevents large concurrent task bursts. Cloud Run had fewer of these issues, but they noted other limitations in the form of an execution time limit or a small resource pool.

Md. Foyzur Rahman [2] developed a web application on AWS, GCP on CaaS and PaaS platforms on docker and evaluated deployment, performance, cost and complexity of service usage. The study includes GCP Cloud Run, GCP GKE AutoPilot, AWS Fargate and AWS EKS, all 3 services are serverless. The aim of the study was to examine whether serverless platforms would be suitable in terms of performance and cost changes. The study looked at these parameters between AWS EKS with AWS Fargate and GCP GKE AutoPilot Avg response time, Minimum response time, median response time, Maximum response time, Total HTTP requests, requests per second CPU Utilization. In the comparison of AWS EKS with AWS Fargate and Cloud Run, Average Response Time, Maximum Response Time, Total HTTP Requests, Request per second, 90th percentile (ms), 95th percentile (ms) parameters were used. According to the data, Amazon EKS outperforms GCP Cloud Run in terms of maximum response time, although GCP Cloud Run has significantly lower average latency. On the other hand, the 90th and 95th percentile latencies exhibit the largest performance disparity. GCP Cloud Run fared better overall. The GCP Cloud Run PaaS platform outperformed the AWS EKS with Fargate CaaS platform in terms of web application performance. The results of the performance test showed that GKE AutoPilot lagged slightly behind both AWS EKS with AWS Fargate and GCP Cloud Run, which both performed rather well. The lone negative aspect of GCP Cloud Run is that users may occasionally encounter a latency for first user queries of more than eight seconds. Price was also compared, and Cloud Run was found to be the most suitable service under certain conditions.

Robert Győrödi et al. [3] presented a comparative study between OnPrem and Azure SQL Server. By developing the restaurant application, they sought to determine how cloud storage and on-premises storage differed from one another. In this study, SQL Server used on a Windows machine in On-Prem was compared with Azure SQL Server. The Windows machine has 32 GB Ram and 1 TB SSD and i7 processor. The comparison was made with SQL queries such as select, update, insert and delete and their performance was measured. They also made a price comparison. They stated that we can conclude that the cloud approach does not bring better performance in all circumstances. Response time data occasionally reveals subpar cloud database performance in contrast to traditional

databases, particularly in the case of the free cloud tier. It was mentioned that cloud is a better choice than on-prem when higher specifications are chosen. The millisecond-to-millisecond disparity between the two methods is sometimes observed. To sum up, cloud computing with flexible pricing plans can frequently provide the finest ways to utilize the cloud's full potential and convenience in terms of availability, scalability, dependability, security, and lastly cost. The study mentioned that large companies (Microsoft, Facebook, Google, Amazon, and others) use the cloud to store data. The study argued that data in the cloud is always convenient to access from anywhere at any time and most importantly it is always secure.

Aman Yevge et al. [4] conducted a study that examined the positive and negative aspects of AWS, GCP and Azure. For AWS, the authors explained items such as Supports all major operating systems, Wide range of service options, Maturity and availability as positive aspects. They mentioned negative aspects such as additional fees for basic services, additional fees for technical support, and a sharp learning curve. As positives for Azure, the authors mentioned different aspects such as integration with Microsoft products, Built-in applications with support for various languages, very low on-demand pricing and Open-source support. On the negative side, they mentioned that it is more difficult to learn than other platforms and weaker in design than other competitors. Positives for GCP include superior scalability, simple installation and configuration, deep discounts and flexible contracts. On the negative side, they cited less diversity of features, fewer service offerings, and fewer global data centers. They also compared 3 different cloud providers with 16 different features, and as a result, AWS scored higher than Azure and GCP, but Azure and GCP provided superior services and features.

Manish Saraswat et al. [5] presented a detailed study on AWS, GCP and Azure, comparing them on a service-by-service basis with different parameters. They evaluated these three cloud providers from different perspectives. Infrastructure computer services, network technologies, storage technologies, database support, backup services, and essential tools are the criteria. according to the specified parameters. The evaluation's conclusion was that AWS was the most sensible option.

Pallavi Wankhede et al. [6] compare the similarities and differences between three cloud providers in this article. They have used some parameters while comparing these environments. The parameters are Specifications, Pricing, Database and Virtualization, and General. As a result, they stated that the choice of which cloud environment to use can vary depending on the user's requirements.

Tabish Mufti et al. [7] presented a pricing comparison for AWS, GCP and Azure. This study examined various cost metrics for price comparison. These metrics are Hardware cost, Maintenance cost, Labor, Computing, storing cost and Sharing Costs. Also, benefits and limitations are given for each cloud provider separately. In conclusion, according to the authors, the three cloud providers have in common on-demand services, flexibility, support, and security.

Xiaolin Cheng et al. [8] presented an extensive performance comparison in order to examine the relationship between cloud pricing and function. They compared eight different cloud providers and grouped them into small and large cloud providers using the market share ratio. According to the authors, most 3 small cloud providers were more consistent and stable than large ones.

Sabiha Nasrin et al. [9] presented a detailed study on AWS, GCP, Azure and Fission and compared cloud environments with 15 different parameters. Some of the comparison parameters are Storage, Security Factor, Language Variation, Virtualization Technique, User Interface etc. They also presented execution time values on AWS, GCP and Fission for the application that calculates Fibonacci numbers. AWS and GCP were run in the same configurations. According to this run, AWS gave the best time in Fibonacci sequence, then GCP and finally Fission. In terms of memory usage, AWS ranked first, with an average memory usage of 47,781 MB. The average memory usage for GCP is 61.8755 MB. In conclusion, based on the 15 parameters used for the comparison and the additional comparison of execution time and memory usage, they concluded that more resources are available to AWS than to the others. Developers have indicated that they are more interested in Amazon than others and therefore emphasized that it has a large developer community. Conversely, Google offers a far greater number

of open-source services and a vast development space. GCP is a better option financially than AWS and Azure. The user benefits from having access to all Google goods with this platform. Windows users have a lot of opportunities with Azure, and the Windows user population is the largest in the world. They therefore underlined how Azure is more suitable for enterprises.

Irfan Bari et al. [10] presented a study for AWS, Azure, Rackspace cloud providers. They compared these 3 cloud environments using cost and performance metrics. While comparing in terms of cost, they evaluated in 3 main sections and obtained separate results for small scale computation, medium scale computation, Large scale computation. The performance metric was based on response time. The authors said that AWS provides the best price option, Azure has an advantage in large-scale companies, and RackSpace is costly in any case.

Srikanth Kandula et al. [11] Presented a detailed study on AWS, Microsoft Azure and Google App Engine, Rackspace Cloud Servers comparing them on a service-by-service basis with different parameters. They chose AWS and Rackspace Cloud Servers because they host the largest number of web services. Google App Engine is a PaaS provider and chose Azure because it has compute and storage services similar to AWS. Their goal is to provide the user with a broad set of metrics and demonstrate the performance cost trade-off. They concentrate on measures including available bandwidth, network latency, storage service response time, CPU, RAM, and disk I/O, scaling latency, and consistency time. They see differences in virtual instances, storage services, and network transfers amongst cloud providers in terms of performance and cost.

Stefan Boneder et al. [12] Presented a detailed study on Amazon Web Services, Microsoft Azure, and Google Cloud Platform, comparing them on a service-by-service basis with different parameters. 3 providers were compared with separate parameters for compute, container, serverless compute, cloud security and IAM. For each provider, the parameters were evaluated and scored, and whichever provider scored the best was highlighted. In the Compute category, AWS Elastic Compute Cloud, Azure Virtual Machine and Google Compute Engine were compared respectively, and Azure Virtual

Machine reached the highest number of points in use cases A and B in the comparison made over 16 metrics. Then a comparison was made in the container category and Amazon Elastic Kubernetes Service, Azure Kubernetes Service, Google Kubernetes Engine were compared. Based on 14 different metrics, Google Kubernetes Engine received the highest score. In the serverless computing category, AWS Lambda, Azure Function and Google Cloud Function were used, compared on 6 different parameters and GCP Cloud Function received the highest score. In the cloud security category, AWS Security Hub, Microsoft Defender for Cloud, GCP Security Command Center were used. According to the comparison, Azure is the most optimal solution. In the IAM category, all providers scored equally when compared over 5 parameters. No provider stood out in this section. In general terms, different providers stood out at each stage.

Martin Wahlberg [13] presented a detailed study on Amazon Web Services and Google Cloud Platform, comparing AWS and GCP using Spark Job, using 5 different measurement parameters and 3 different metrics. The metrics are CPU intensive, I/O intensive and MapReduce. In the CPU intensive comparison, AWS finished ahead in 5 different machine types, CPU utilization, memory utilization and completion time were observed. AWS was the provider with the lowest completion time. In the I/O intensive benchmark, AWS and GCP were ahead in 5 different metrics. In the comparison, disk utilization, memory utilization, CPU utilization and completion time were looked at. In the MapReduce comparison, CPU utilization, memory utilization, disk utilization and completion time were looked at and GCP had the lowest Completion time in all 5 different types. In summary, in terms of performance, AWS was ahead in all types except type 2 for CPU Intensive. For I/O Intensive, both AWS and GCP are successful. For MapReduce, GCP was completely successful. In terms of cost, AWS was evaluated as less costly than GCP.

Muhammad Ayoub Kamal et al. [14] presented a detailed study on AWS, GCP and Azure, comparing the storage, compute, and infrastructure service characteristics of cloud environments. They mentioned that these features are effective when choosing between environments. As a result of their study, they mentioned that AWS is the

worldwide leader and ranks first in market share, while GCP and Azure are better than AWS in terms of security and pricing.

Hyungro Lee et al. [15] conducted a study comparing serverless computing services such as IBM Cloud Functions (Apache OpenWhisk), Microsoft Azure Functions, Google Cloud Functions, and Amazon Lambda in terms of concurrent calls. The results show that the flexibility of Amazon Lambda is superior to the other services used in terms of CPU performance, network bandwidth and file I/O throughput when making concurrent function calls for dynamic workloads. The authors also concluded that serverless computing would be more cost-effective than computing on traditional virtual machines. They advise users of traditional virtual machines to migrate to serverless systems.

Prerna Jain et al. [16] presented a study of services on Amazon Web Service. In this study, AWS Lambda, ECS using EC2, ECS using Fargate services were used. The performance was measured by using Start Render Time, Load Time, First CPU, Speed Index Idle parameters. NodeJS application was tested on various performance metrics. Fargate shows the best performance compared to both. The authors mention that the idea of using a serverless Container-based execution platform as a service will provide better performance in the future.

Roberto Vera Alvarez et al. [17] provided a breakdown of the best practices, expenses, and complexity involved in carrying out the main workflow components. The study made use of AWS and GCP. Similar computational instances with 16, 32, and 64 vCPUs were tested for each cloud provider. The results show that when it comes to instance creation, setup, and publishing, the AWS platform outperforms GCP. Despite the differences in the configuration and setup of batch systems between GCP and AWS, the cost and processing time are similar for the type of workflow designed. The authors noted that GCP is more user-friendly to use by only needing a JSON file for batch operations. On the other hand, AWS requires a full installation of all batch processing system components. Another important detail mentioned in the study is the following: For routine data analysis tasks in research labs, GCP provides a better option. However, AWS operates more effectively when set up correctly when it comes to the operations of

creating, deploying, and releasing machines. ECS provides the advantage of reusing samples by reducing the cost in big data analysis projects. AWS offers a more suitable platform for big data analytics groups to create queuing and computing environments for multiple pipelines.

In this article, Robert Kelley et al. [18] provide a comparison of the various computing resources offered by AWS, Azure and GCP. The choice of platform depends on a large number of variables. They used many variables, including geographic availability, security and compatibility, operating system support, container support, and serverless computing support. The authors emphasized that, overall, they found that for computing resources, Microsoft Azure was the preferred provider for most of the characteristics we identified. However, this difference was quite small. Some limitations of the study were mentioned, for example, that more features could have been used, and that there was no cost analysis.

Niranjan S et al. [19] compared GCP Cloud Run and Azure App Service. With a sample size of 20, the study ran and evaluated novel Microsoft Azure-based Docker containers as well as GCP-based containers for varying numbers of runs. The maximum allowable error was set at 0.05 and the minimum power analysis was placed at 0.8 for the G-Power computation. The same set of 20 containerized apps was used to determine how long it took for each platform's application to deploy in order to determine how accurate each platform was. The authors state that the Google cloud platform-based Docker container was determined to be inferior to the Microsoft Azure-based Docker container. The study involved statistical computations of mean, standard deviation, and standard error mean. When comparing deployment times, Azure App Service outperforms GCP Cloud Run.

Luis Herrera-Izquierdo et al. [20] in his study has included the differences and some comparisons between virtual machine and docker. CPU performance test, RAM performance test, network performance test and disk performance tests were measured between VM and Docker. In CPU performance, Docker outperformed VM configuration in all three scenarios. In the RAM test, the Docker container outperformed the VM by 137

MB/s, which is 4.5% better utilization of resources. In the Network test, Network performance is 12.6 times higher in the container in client mode; in server mode, the container was measured to be 4.5 times faster than the VM by the authors. In disk tests, the container VM was found to have a lower throughput compared to the test results. The VM is clearly faster in file processing. In general, the results of the tests show that Docker containers are more advantageous than VirtualBox in terms of efficiency. Containers are reported to use the available hardware resources more efficiently, especially in terms of networking. However, the VM performed better than the container in terms of read and write performance. The authors concluded that for software applications with very little I/O, using containers provides better performance.

In this article, Joseph Mart et al. [21] provide an overview of AWS, GCP and Azure. They evaluated the cloud providers on issues such as security, reliability, performance, cost optimization, sustainability and stated their opinions. They gave general information about cloud providers.

Thameez Ahmad Bodhanya [22] compared container platforms in the AWS environment using certain criteria in his article. The aim of this project is to investigate and compare cloud container orchestration platforms. These metrics helped in the comparison. ease of adoption and configuration, deployment process, restrictions and limitations, performance, cost impacts, reliability, and resilience. AWS Lambda, AWS ECS - EC2, AWS ECS-Fargate, AWS EKS-EC2, AWS EKS-Fargate were used in the study. CPU performance was evaluated by using a program that calculates prime numbers. According to the results, AWS EKS-EC2 gave the best time. EKS-Fargate ran 20% slower. ECS supported services were about 50% slower. The two EC2-powered instances completed about 9% (EKS) and 10% (ECS) fewer instructions respectively. The two Fargate-powered instances struggled to complete this benchmark, with the EKS instance completing 80% fewer tasks and the ECS instance completing 95% fewer tasks. Average maximum speeds were measured and the EC2-powered EKS instance outperformed the baseline RKE instance, completing tasks about 36% faster. This was followed by the significantly slower fargate-powered EKS instance and the EC2-powered ECS instance,

both of which completed tasks close to 80% slower than the baseline. Finally, the fargate-assisted ECS instance completed the benchmark close to 88% slower. After all these results, the authors offer some conclusions. Workloads running EKS generally performed closer to the benchmark set across the widest range of tests, while both ECS and then Lambda consistently lagged behind with low but measurable performance impact. EC2-based workloads consistently performed closer to the benchmark set, while Fargate and Lambda underperformed even under the same orchestration platform.

Aditi Rajan Khot [23] studied AWS, GCP, Oracle, and Microsoft Azure environments. The aim of this paper is to introduce the concept of multi-cloud computing to leverage the advantages of different cloud providers and maximize their benefits in a single network architecture. Comparisons of the above-mentioned 4 environments are given. For the 4 environments, service information is given based on storage and database, compute, network and security. The multi-cloud structure is evaluated in the study. In conclusion, the author emphasized that AWS has the most comprehensive services and global reach, while Azure may be the best choice if the servers are Windows and Microsoft software. If a small, web-based startup aims for rapid scale-up, GCP is the most suitable provider. With its cheap pricing model, Oracle software and autonomous databases, Oracle Cloud Infrastructure is very suitable. He expressed concern about multi-cloud and warned that it can create problems if left unchecked. However, he also mentioned the positive aspects of multi-cloud. The author mentioned that the power of a multi-cloud strategy can help an organization to maintain its flexibility.

Anoop Abraham et al. [24] aim to compare the performance and availability of a containerized mobile-web application on serverless and server-based services on Google Cloud. Cloud Run, App Engine, Compute Engine are used in the study and comparisons are given on the basis of service configurations, pricing and features, as well as performance comparison for these 3 services and comparison is done using metrics such as latency, throughput, container startup. The latency and throughput of the application hosted on these three different platforms were measured. The results show that Cloud Run is a highly efficient and cost-effective option for deploying real-time web applications.

The authors found that Cloud Run can save more money compared to other GCP services. Cloud Run has the ability to auto-scale according to load, providing low latency and good throughput performance. As a result, Google Cloud Run has become a viable option for real-time web application implementations. The authors said that the work could be extended to AWS and Azure in the future.

Alireza Goli et al. [25] did a study on migrating from monolithic architecture to serverless architecture, case study in the Fintech domain. serverless architecture was compared to the old monolithic version in terms of performance, scalability and cost. Cloud Run was used for serverless architecture. Each instance of the service is allocated 2 GB of memory. In the study, the monolithic version was run on GCP in two separate virtual machines with different characteristics. The first virtual machine has 4 vCPUs and 15 GB memory, while the second has 96 vCPUs and 360 GB memory. Both systems were benchmarked on a set of 100 documents of 7 to 15 pages. In the second experiment, each experiment was repeated three times and the result of the first run was ignored to eliminate the cold start effect. According to the first experiment, the serverless architecture was about 93 times faster than the monolithic architecture. In terms of cost, the monolithic architecture turned out to be about 49.38% cheaper than the serverless architecture. In conclusion, both experiments show that while the serverless architecture significantly improves performance, it only leads to a marginal increase in cost.

Juan Mera Menendez [26] in this work compared 9 prototypes of the same microservice application using different services. Amazon Web Services was used and the impact of using different technologies such as AWS ECS Fargate, AWS Lambda, DynamoDB or DocumentDB was analyzed by measuring both cost and performance. When the 9 prototypes were compared in terms of cost, AWS Lambda was the most cost effective, followed by AWS Lambda+DynamoDB, with AWS Fargate being the third best. In terms of performance, a stress test and a performance test were conducted, and AWS EKS + DynamoDB gave the best result in both categories. AWS Fargate was not in the top three.

In this work, Ikshaku Goswami [27] discusses the development of machine learning inference and training architectures using serverless services from AWS and GCP. These architectures are evaluated based on the cost and time to train or execute. While the focus of the study is on exploring serverless services, the study also deals with machine learning tasks critical for building a model. The study compares Lambda and GCF, emphasizing that Lambda's cost is generally slightly lower when calculated overall, but GCF will allow a slightly higher number of calls. The cold start problem of the services is also mentioned in the study. AWS Elastic Container Service with Fargate is also mentioned in the paper. In this article, Fargate is discussed in detail and compared with Lambda in terms of execution times. The equivalent of Fargate in GCP is Google Cloud Run. The authors emphasize that Fargate enables developers to utilize AWS' serverless platform to its maximum benefit. The properties of Lambda and Fargate are given and Fargate is described in the article. It is observed that the average execution time for an inference task is about 600 ms for a single container instance. As the number of invocations increased, Fargate started spinning more containers to handle the traffic, which further reduced response times. The author compared AWS and GCP and compared Lambda and GCF. They conclude that serverless architecture for inference-based architectures can significantly reduce the running cost for a machine learning model. Lambda functions or GCF can be used in combination with external storage systems such as EFS to provide a scalable platform based on low latency. However, in this research, it can be concluded that based on response times and running cost, serverless containers such as ECS may be better to run on Fargate.

Dheeraj Chahal et al. [28] presented comparative results by running their deep learning model (OCR API) on AWS Lambda and On-Prem separately. The purpose of this study was to evaluate the performance and cost of a serverless platform and compare it to on-premises and cloud VM deployment. AWS Lambda was used in conjunction with EFS and EC2 was also used in the study. On-premises, EC2 virtual machine, AWS Lambda response times were compared, and AWS Lambda gave the best results. Lambda response time was slightly lower than EC2 VM and on premise for small jobs. As the workload increased, Lambda response time remained constant while the response times

of other competitors increased. Lambda auto scaled. For deploying applications with large models, the approach of using EFS with Lambda proved to be more suitable. The authors also noted that while using EFS and Lambda together increases the cost, it is still less than using a VM in the cloud, and performance metrics such as throughput and response time using Lambda and EFS are better than using a large VM. On the negative side, they cautioned that cold start can result in a high response time and that FaaS may not be a good option when real-time demands are sparse. They emphasized that it is difficult to estimate the cost in serverless architecture. As a result, they concluded that serverless architecture significantly improves performance at lower costs.

Olli Paakkunainen [29] investigated the feasibility of a web application backend on FaaS platforms. The investigation was carried out by examining backend solutions in terms of performance, cost and development experience. The author conducted multiple synthetic performance tests to examine runtime and cross-platform differences. Costs are analyzed for different FaaS runtimes and providers. FaaS infrastructure costs are compared with PaaS platforms. The study includes a comparison of AWS Lambda, Azure Functions, Google Cloud Functions, IBM Cloud Functions, and cost comparisons of these services. The author also included open source FaaS platforms such as Fission and Kubeless. Incremental load test performances and spike load tests were performed using 2 FaaS platforms and 1 PaaS service. These services are AWS Lambda, Google App Engine and Google Cloud Functions. The author gave the cost comparison of these services. Google App Engine was the most cost effective. Based on the results, the company decided to use PaaS instead of FaaS to implement the web application.

Akash Chauhan [30] discussed different aspects of AWS, Azure and GCP environments. The aim of the paper is to help companies choose the most suitable platform for their needs. The study focuses on the most important aspects of each platform, such as the virtual machines they offer, the storage options they provide, the database services they provide, and the serverless computing capabilities they provide. In addition, cost, availability, security, and scalability are also addressed. The positive and negative aspects of these environments are also given. Key features for all 3 platforms are given. Then

these environments are compared by looking at features such as cost, performance, and features. In conclusion, the author emphasized that Amazon, Azure and GCP are reliable cloud computing platforms and which platform to choose will depend on the specific needs and demands of each company.

None of the aforementioned papers apply to a scenario where both serverless and on-premises systems are used together. Looking at the literature, I could not find a study where both serverless systems and on prem systems are combined. Looking at the studies in the literature, there are studies showing performance comparisons of AWS ECS-Fargate, GCP Cloud Run services. In the literature, general comparisons of cloud providers such as AWS, GCP, AZURE are also described and comparison of these environments using different parameters is given. There are such studies in the literature, but in this study, Windows was used in addition to cloud services and a comparison was made using different parameters such as memory usage, CPU usage, network and response time etc. using a web application as a reference. Therefore, I believe that this study is the first in this sense and I believe that it will bring a different perspective to literature, take its place in the literature and close this gap. In this work, I specifically investigate a scenario where an ML model deployed on Docker is tested on both serverless architecture and on-premises systems. I aim to address this specific issue and compare the performance of the ML model under specific workloads in both serverless and on-premises environments.

c) Overview

The paper is organized as follows. In Section 1, I'll provide an overview of the technologies. In Section 2, the complete recommended container-based design is presented together with details about the cloud providers. Section 3 presents the experimental progress and evaluation criteria. In Section 4, the benchmark findings are explained. Section 5 covers the topic and evaluates related works as well as the state of the art at the moment.

1. BACKGROUND

1.1 Cloud Computing

Cloud Computing is the presentation of all systems such as servers, storage, databases, and machines on the Internet in the cloud environment. These systems can be used in any environment where there is internet. For computers and other devices, it can be used at any time. Without the need for installation, you can keep data on a virtual server, access it from anywhere and access it from any device. When you have a cloud infrastructure, you can save more on storage space, transfer data faster and save money. It helps save time and labor for businesses of different sizes. Day by day, most companies around the world move their data to cloud computing infrastructure and benefit from these advantages.

1.1.1 Benefits of Cloud Computing

- Reduces costs.
- Eliminates infrastructure complexity.
- Expands the workspace.
- Protects your data on high security cloud servers.
- There is always access to information.

1.1.2 Types of Cloud Computing

Public Cloud: Systems such as servers, storage, databases are offered to users by cloud providers. These services can be free or charged according to usage. Small and medium-sized companies use a pay-as-you-go model. Customers pay for CPU (central processing unit), storage and bandwidth.

Private Cloud: Private cloud known as internal cloud and enterprise cloud. A private cloud is used only by a company. It can be located in the data center of the company or cloud provider. A high level of security and control is important. International companies and government organizations may prefer this structure. Microsoft's Hyper-V and System Center products are examples of Private Cloud.

Hybrid Cloud: Public and private clouds are combined to create hybrid clouds. Public clouds make more sense in situations where security can be maintained to a minimum, even though private clouds are utilized where security and privacy are more crucial. The way these two structures are combined also differs according on how big the companies are. In terms of cost, small companies are more likely to choose public cloud, while companies with high data size prefer hybrid cloud.

Community Cloud: Community Cloud is a cloud technology that hosts shared services with a few companies, much less preferred than other technologies.

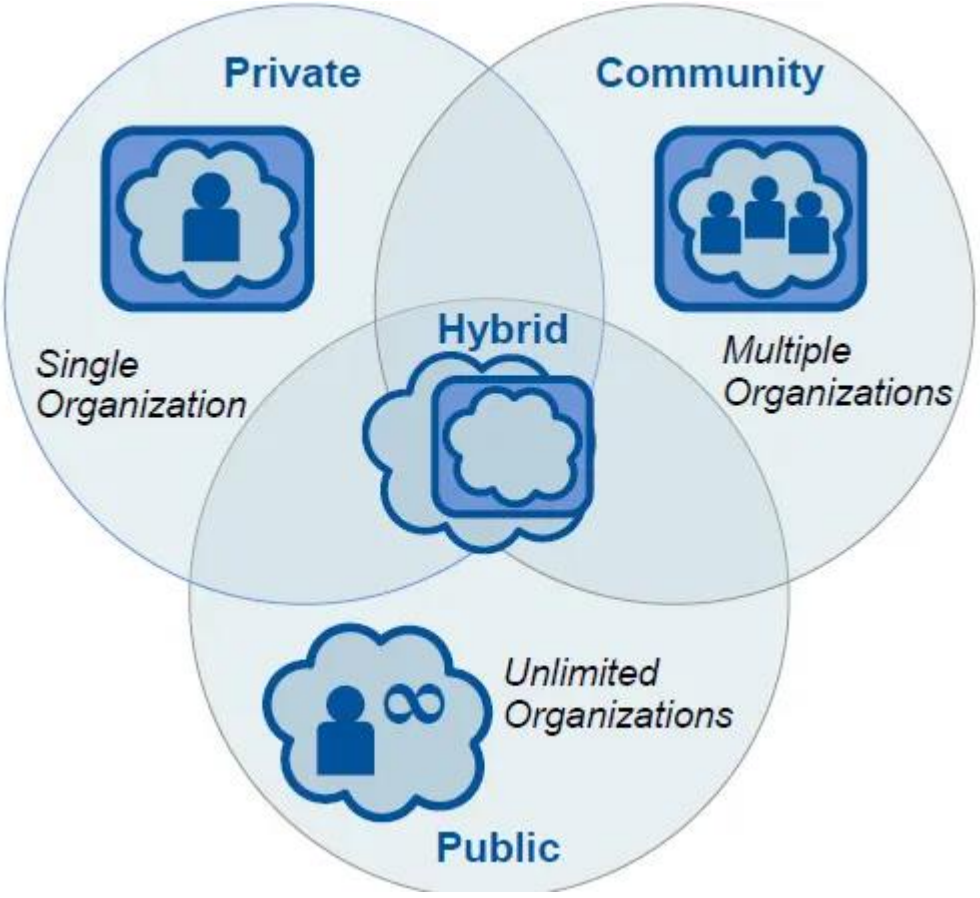

Figure 1.1: Type of the Cloud Computing [31]

1.1.3 Cloud Computing Service Types

Infrastructure as a Service (IaaS): Infrastructure as a Service is a kind of pay-as-you-go cloud computing service that provides network, storage, and processing capabilities. IaaS saves the cost and reduces the complexity of setting up physical servers and data center infrastructure. With IaaS, you use the resources you need and pay as you use them.

PaaS (Platform as A Service): Platform as A Service is a system where users develop applications in different programming languages on the direct cloud without dealing with systems such as operating systems and networks. It allows customers to manage mobile and web applications.

SaaS (Software as A Service): With software as a service, you may use a web browser to view a program without needing to download it to your computer. It is a cloud-based service. The customer accesses the application software over the internet. The service subscription is utilized on a pay-as-you-go basis.

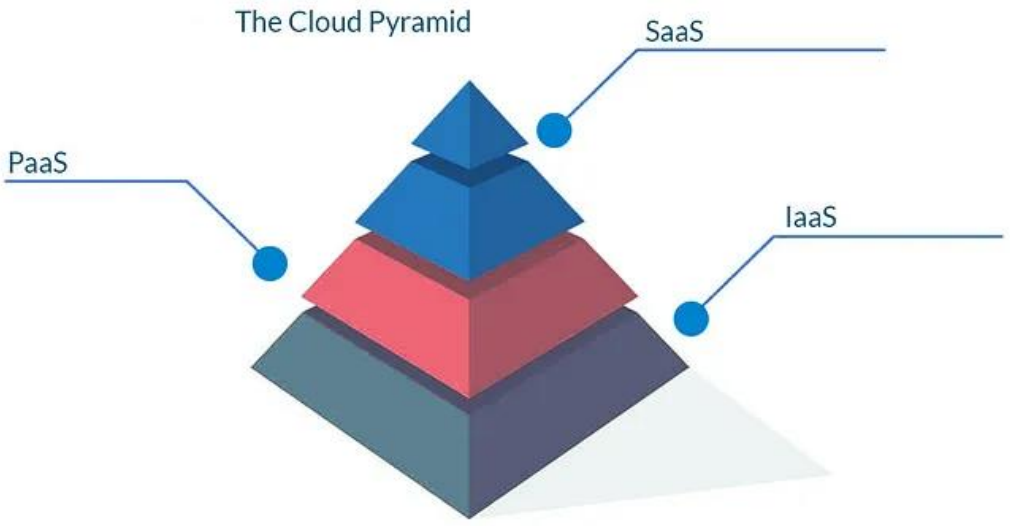

Figure 1.2: Service Type of the Computing [34]

1.2 On-Premises

On-Premises systems are better than cloud systems in terms of security because the data is kept in your own data centers and the on-premises systems are managed by the IT team and are completely unique to your company. It does not require an internet connection for user access. One of the biggest disadvantages of On-Premises software is that the labor and cost to manage the system is higher than cloud systems. IT employees are needed for problems that arise in the systems.

1.2.1 Advantages of On-Premises

- Long-term costs are lower.
- The user retains full control.
- Allows the implementation of its own security policies.
- Does not require internet access.

1.2.2 Disadvantages of On-Premises

- Invest in hardware and infrastructure.
- In-house IT team is needed.
- The system cannot be accessed on the move.

1.3 Machine Learning

The vast area of artificial intelligence encompasses deep learning and machine learning and was first introduced by Alan Turing in 1950 with the concept of "intelligent" and "mindless" machines. Machine learning is a sub-branch of AI that learns and improves performance based on the data it is exposed to. Machine learning applications evolve with use and start to give more accurate results when there is more data access. It is possible to see machine learning studies in different sectors. These sectors are in many sectors such as automotive, health, commerce. Supervised learning, unsupervised learning, and mixed learning are the three types of machine learning approaches. focused on reinforcement of learning. In supervised learning, which uses classification algorithms, the machine is aware of the target value to be learned and adjusts its learning accordingly. Unsupervised

learning involves training based on data without labels or a specific, defined output, this type of learning involves clustering algorithms. In reinforcement learning, it helps determine the outcome based on a feedback loop, there is a reward system. The more rewards, the better the system learns. Machine Learning projects are developed in 7 steps. These steps are indicated in the photo below.

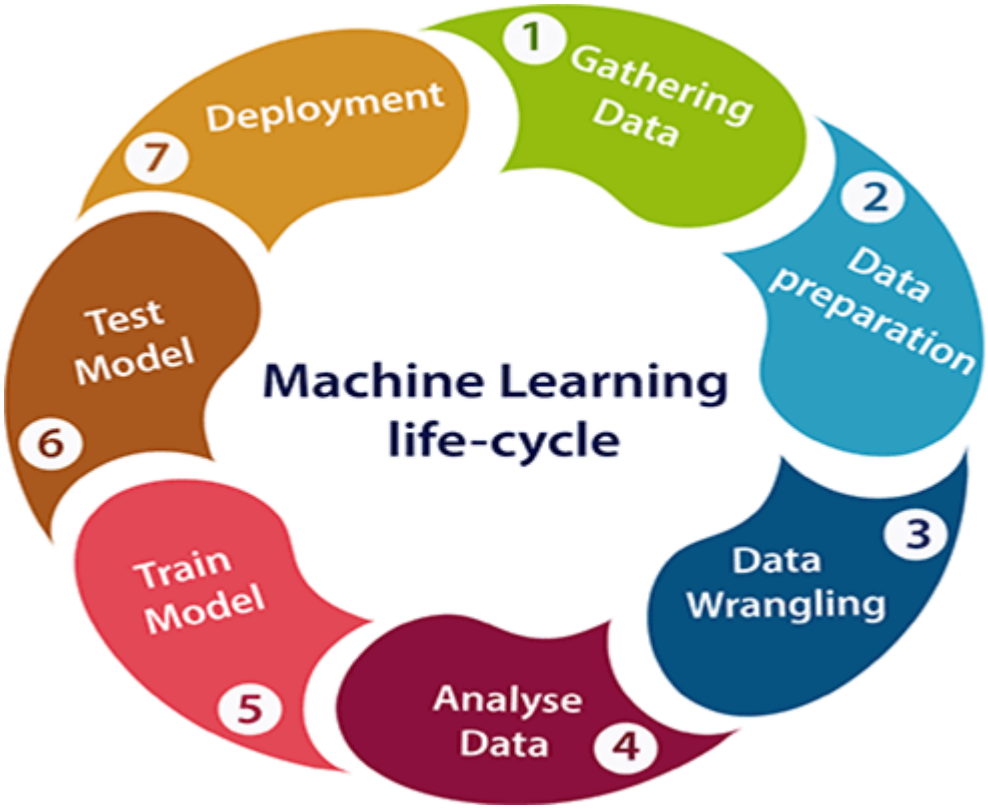

Figure 1.3: Machine Learning Lifecycle [32]

When you want to deploy your developed ML Model, the trained model is output in pickle or md5 format, then this model is deployed on different platforms and made ready for use, it can be deployed on on-premises systems or cloud systems.

1.4 Docker

Docker is a container environment developed with open source, where we can perform operations, such as developing and running applications on an isolated environment. Since containers are isolated from each other, different applications can run.

Docker containers are lighter, faster and more resource efficient than virtual machines. Unlike a virtual machine, Docker provides isolation by using container structures on a single operating system instead of creating a separate virtual operating system for each application. This also provides a performance boost.

1.4.1 Docker Image

Containers consist of layered images, docker images are the image files of the applications to be installed in the containers. For example, python, MySQL image files can be considered.

1.4.2 Docker Registry

Developers like GitHub can share open source docker images on Docker hub and other developers can download and use these images. The images are kept in the docker registry.

1.4.3 Advantages of Docker

- Faster scaling of existing systems,
- Versioning is more efficient with containers,
- Running multiple workloads with less resource consumption,
- Standardizing environments to ensure consistency in the software development cycle,
- It is more cost-effective than virtual machines.

1.4.4 Docker vs Virtual Machine

VM (Virtual Machine)

- OS: Full operating system
- Insulation: High
- Becoming operational: Minutes
- Versioning: None
- Easy shareability: Low

Docker

- OS: Minimized operating system image
- Isolation: Lower
- Becoming operational: Seconds
- Versioning: High
- Easy shareability: High

1.5 Serverless Architecture

Serverless means that you run the necessary code on the services without having to think about the server. IT teams not need to be concerned about managing servers and can focus on writing the code. The cloud provider manages the entire infrastructure on its own and automatically provides the user with the resources needed to execute the code. Serverless architecture allows developers to deploy their applications in a faster, scalable and cost-effective way.

1.5.1 Advantages of Serverless Architecture

- There is no Server Maintenance
- Has an Auto Scaling feature.
- Reduces time-to-market because infrastructure and server settings are not involved.
- Low operating costs.
- Latency through the application is low.
- There is enhanced flexibility over the application.

1.5.2 Serverless Architecture Services on Cloud Providers

Table 1.1 Serverless Architecture Services on Cloud Providers

AWS	**GCP**	**MS Azure**
AWS Lambda	Cloud Run	Azure Container Apps
AWS Fargate	Cloud Function	Azure Kubernetes Service
Amazon EventBridge	Eventarc	Azure Functions
AWS Step Functions	Workflows	Azure App Service
Amazon SQS	Firestore	Azure Devops
Amazon SNS	CloudSQL	Azure SQL Database
Amazon API Gateway	CloudSpanner	Azure Cosmos DB
AWS AppSync	CloudScheduler	Azure Blob Storage
Amazon S3		
Amazon DynamoDB		
Amazon RDS Proxy		
Amazon Aurora		
Amazon Redshift		

1.6 Python

Guido van Rossum is credited with developing the Python programming language. It is a multipurpose language with an interpretable, modular, and object-oriented structure. Python is known for its high readability and simplicity. For this reason, it is highly preferred for beginners and also reduces the effort of programmers and increases their productivity.

1.6.1 Uses Cases of Python

Program Development: You can write desktop, web applications and games with Python. For web development, Flask and Django are utilized.

Data Science and Engineering: Many people use Python for data science and analysis. The most used libraries are Pandas, NumPy and SciPy.

Artificial Intelligence Machine Learning: Python is widely used for artificial intelligence machine learning. Libraries like TensorFlow and PyTorch are used in this field.

Web Scraping: Python is also used to pull data from a website. BeautifulSoup and Scrapy libraries are used in this area.

Game: Game development with the Pygame library.

Scripting: It is used to automate jobs by writing automation scripts.

2. CLOUD PROVIDERS AND ON-PREMISE SYSTEM

2.1 Google Cloud Platform

Google provides the Google Cloud Platform. It's a cloud environment that utilizes the same infrastructure as end-user apps like YouTube and Google Search. Google Cloud Platform offers various cloud services, including computing, data storage, data analytics, and machine learning, in line with the way technology advances and the world changes. Google announced its first service, App Engine, in April 2008 and it became generally available in 2011. GCP runs these services for users in its own centers. Charges are based on the amount of usage of these services (pay as you go). There are over 100 services on the GCP, available at various locations in North America, South America, Europe, Asia, and Australia. Each location has its own pricing, with data centers in the Americas generally charging less. Some of the most important services available on the GCP are categorized separately below.

- Compute
- Networking
- Storage and Databases
- Big Data
- Machine Learning

Compute: GCP offers a choice of self-scalable virtual machines that can be customized according to the needs of the users, you can deploy the written code directly or with the help of containers.

- Google Compute Engine
- Google App Engine
- Google Kubernetes Engine
- Google Cloud Container Registry
- Cloud Functions

Networking: This includes network related services, for connection to different cloud environments or systems, these services can be used for security.

- Google Virtual Private Cloud (VPC)
- Google Cloud Load Balancing
- Content Delivery Network
- Google Cloud Interconnect
- Google Cloud DNS

Storage and Databases: GCP includes NOSQL and Relational databases and also has a storage service that can hold Supervised, Unsupervised and semi-supervised data.

- Google Cloud Storage
- Cloud SQL
- Cloud Bigtable
- Google Cloud Datastore
- Persistent Disk

Big Data: GCP has services to analyze, transform and transfer data on Big Data.

- Google BigQuery
- Google Cloud Dataproc
- Google Cloud Datalab
- Google Cloud Pub/Sub

Cloud AI: Provides users with different services for working in the field of machine learning.

- Cloud Machine Learning
- Vision API
- Speech API
- Natural Language API
- Translation API

- Jobs API

2.2 Amazon Web Services

Amazon Web Services (AWS) is a cloud service platform with services such as compute and database storage and was founded in 2006. By establishing server farms in important locations around the world, it tries to produce a solution to every problem of users, AWS currently serves in approximately 30 regions and approximately 100 different availability zones. In 2020, its competitors Microsoft, Google and IBM have 20%, 9% and 5% market share respectively, while AWS has a 32% share among all clouds. AWS supports its users at many points such as virtual servers, storage, databases, content distribution, artificial intelligence and machine learning, network services.

Some important services on AWS are categorized separately below.

- Compute
- Networking
- Storage and Databases
- Big Data
- Machine Learning

Compute: Supports different compute models, from Virtual Servers to serverless compute execution. For developers, virtual servers can be deployed on AWS, helping them meet their compute power needs.

- Amazon EC2 (Elastic Compute Cloud)
- AWS Lambda
- Amazon ECS (Elastic Container Service)
- Amazon Lightsail

Networking: Helps AWS users build and manage virtual networks, DNS services, traffic balancing and network infrastructure

- Amazon Virtual Private Cloud (VPC)

- Amazon Route 53
- AWS Direct Connect
- AWS Elastic Load Balancing

Storage and Databases: AWS has services for storing all kinds of data, structured and unstructured data can be stored, and services for relational databases or NOSQL databases are also offered.

- Amazon S3 (Simple Storage Service)
- Amazon RDS (Relational Database Service)
- Amazon DynamoDB
- Amazon EBS (Elastic Block Store)

Big Data: For the ever-increasing data, AWS has powerful services for processing, transforming, and analyzing data. It allows you to manage tools such as Hadoop and Spark in big data projects.

- Amazon EMR (Elastic MapReduce)
- Amazon Redshift
- Amazon Kinesis

Machine Learning: There are services for Machine Learning and artificial intelligence studies, development, training, and deployment of machine learning models are developed end-to-end with the services available in AWS.

- Amazon SageMaker
- Amazon Rekognition
- Amazon Comprehend

2.3 Cloud Run

With the help of Cloud Run, a managed computing platform, you can run containers directly on Google's infrastructure. You can upload your code to Cloud Run if you know how to make it into a container image. If you are using programming languages

like Go and Node js, Python, or Java, you have the option to use source-based deployment. This option automatically creates a container for you using the recommended methods for the specific language you are using. Google has created Cloud Run to work seamlessly with other services on Google Cloud, allowing you to create advanced applications.

In simpler terms, Cloud Run lets developers focus on coding rather than spending too much time managing and adjusting their Cloud Run service. You don't need to make a group or handle technology stuff to work efficiently with Cloud Run.

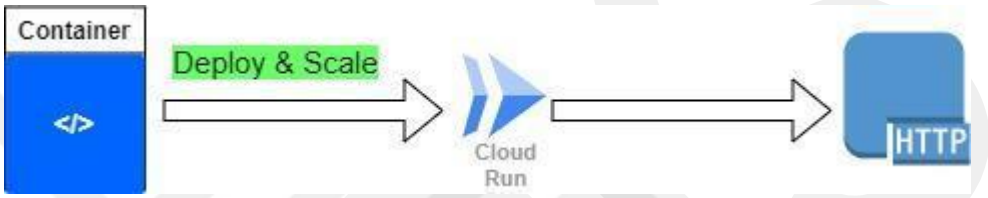

Figure 2.1: Example of Cloud Run Architecture

2.3.1 Features of Cloud Run

- Any Language
- Fully Managed
- Auto-Scaling Feature
- Built on the Knative
- Custom Domains
- Developing Experience
- Redundancy

2.3.2 Pros of Cloud Run

- Container to production in seconds
- Each service is created using a Docker image.
- Scale without a server depending on the number of requests.
- Cloud Run supports all kinds of programming languages thanks to docker.
- It can be integrated into Google Kubernetes Engine clusters using Anthos on GCP.
- It has a simple payment model; you pay as long as the code works.

2.3.3 Cons of Cloud Run

- Certain features that are accessible in Cloud Functions, such as Firestore, are not accessible in Cloud Run.
- Not helpful for apps that do things in the background.
- This can make the application more complicated, making it difficult to understand how it is organized.

2.4 Google Cloud Container Registry

The Google Cloud Container Registry is a place where developers can manage the container images they make with Docker. Here you can analyze vulnerabilities and decide what can be accessed by individuals with specific access control. Google Cloud Container Registry is a tool that allows you to store and manage your software containers. It has certain qualities and advantages that include:

- Google Container Registry is a safe storage service for Docker images that can only be accessed by authorized people. With this service, you can get images fast. You have power to decide who can see or save container images. This control is managed using the registry.
- With Google Container Registry, users can easily turn their code from Cloud Source Repositories, GitHub, or Bitbucket into containers and store them in their Google registry. These containers that are officially recognized can be used directly from various Google services like App Engine, Google Kubernetes Engine, Cloud Run, Cloud Functions, or Firebase with Google Cloud.
- Google Container Registry facilitates the detection of vulnerabilities and provides support to ensure that container images are distributed securely. The threat database provided is constantly updated and remains secure against malware.
- Images that are perceived as risky are not allowed to be distributed and used through Google Kubernetes Engine.

- Google Container Registry uses docker commands as standard, which provides advantages for users in environment transitions and easy adaptation. Container Registry and Docker are fully compatible.
- Container registry supported by Google is available in many regions around the world. Google has data centers in Europe, Asia, America and the Middle East and Container Registry can be used in these regions.

The architecture below represents the architecture implemented on the GCP in this project.

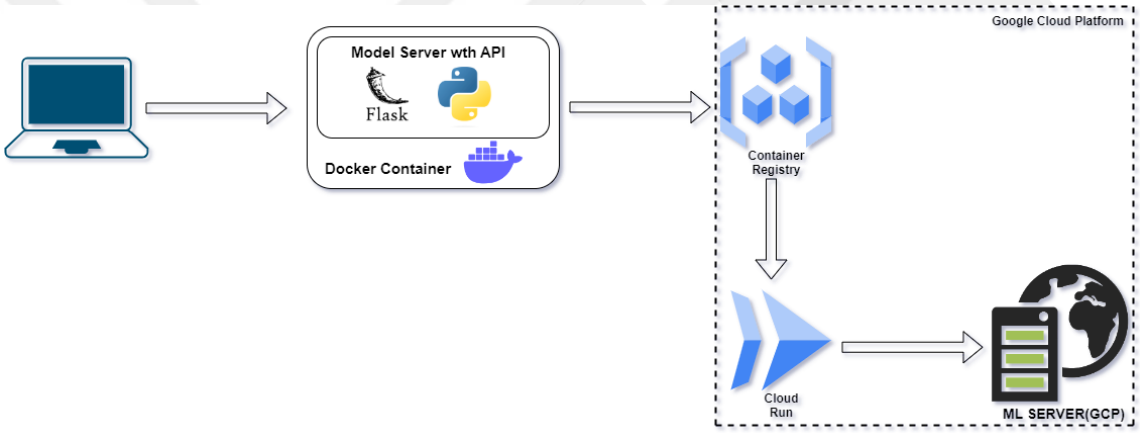

Figure 2.2: GCP Architecture

2.5 AWS Fargate

AWS Fargate creates the opportunity for users to run containers without the cost of container management, creation or scaling optimization. AWS Fargate is a compute engine. With Fargate, users are freed from the necessity of using EC2. You can focus on planning and developing your application instead of dealing with infrastructure maintenance. When running a docker application on Fargate, it will be enough to dockerize the application, select memory and CPU requirements, and create IAM policies. AWS Fargate also makes it easy to scale your applications. AWS Fargate promises high availability of applications and provides the infrastructure. Manages containers in cooperation with Amazon ECS and EKS.

2.5.1 Pros of AWS Fargate

- Better Security
- Reduce Complexity

2.5.2 Cons of AWS Fargate

- Less Customization
- Not cost-effective for small workloads

2.6 AWS Elastic Container Registry

It is a fully managed service on AWS that enables the sharing and distribution of container images and records. AWS ECR has direct connections to multiple services within AWS. It is possible to run your own container repositories with Amazon ECR.

2.6.1 Advantages of AWS Elastic Container Registry

- AWS ECR has a fully managed system.
- Amazon ECR automatically replicates container images across multiple domains, ensuring high availability and durability.
- Amazon ECR can scale Container registrations, which ensures smooth operation and optimal performance for applications running in containers.
- Amazon ECR can work with AWS Identity and Access Management (IAM) to control permissions assigned to resources. It also offers powerful security features such as encryption and image vulnerability scanning.
- Amazon ECR integrates seamlessly with many AWS services such as Amazon Elastic Container Service (ECS) and AWS Lambda, so ECR provides high availability.

The architecture below represents the architecture implemented on the AWS in this project.

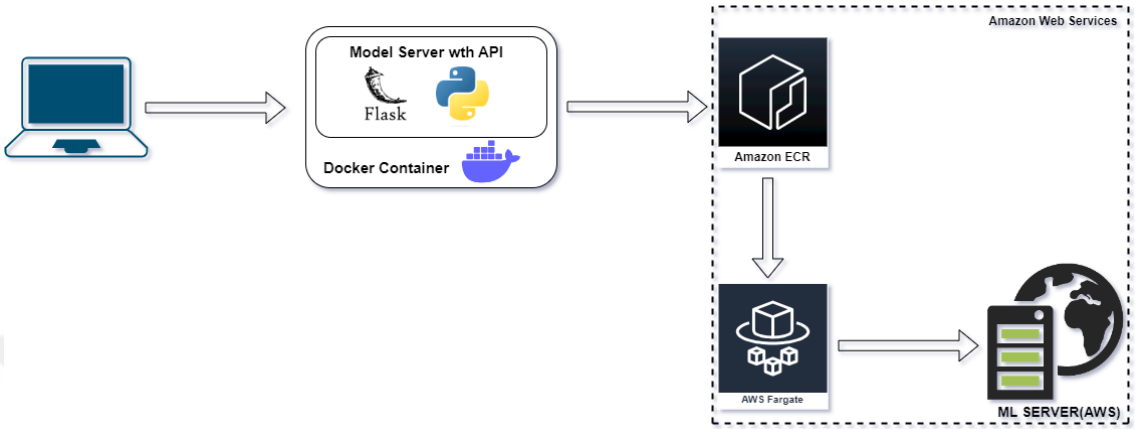

Figure 2.3: AWS Architecture

2.7 Windows

Windows is a group of operating systems produced by Microsoft. It is a server operating system that incorporates features such as data storage, data transfer and corporate network management. The first version of Microsoft Windows was version 1.0, made in 1985. Windows has a graphical user interface (GUI), multitasking features, virtual memory management, and support for a number of peripheral devices. It comes in 32- and 64-bit versions. Client and server editions are composed of Windows operating systems. Client versions that are widely known include Windows 98, ME, XP, Vista, 7, 10, and 11. Windows NT Server, 2000 Server, 2003 Server, and Server 2008 R2 are a few of the Windows server variants.

2.7.1 Pros of Windows

- The interface is user-friendly and easy to learn for beginners.
- Hardware drivers are quick to install and up to date.
- Automatic or on demand windows updates
- Guaranteed long-term support.
- Integration with other windows products.

2.7.2 Cons of Windows

- High license fees
- Security related system error
- More vulnerable to malware than Linux

The architecture below represents the architecture implemented on Windows in this project.

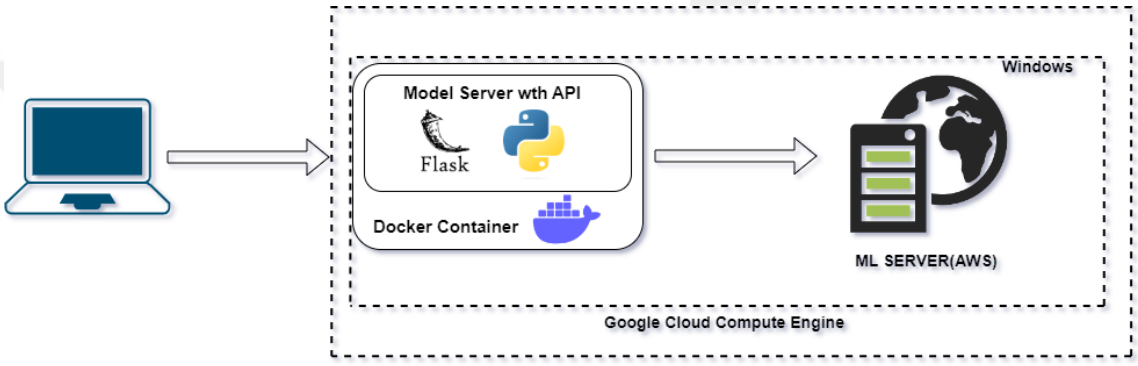

Figure 2.4: Windows Architecture

2.8 Test Details and Measurement Methods

In this part of the thesis, the metrics used in the study and how the services were tested will be discussed. A series of tests and analyses were performed to compare the performance of a Machine Learning (ML) model running on Docker in GCP Cloud Run, AWS Fargate and On-Premises (Windows) environments. Below, the methods and details of the testing process are described.

Data Collection and Preparation Methods: The ML model used for this study is a pretrained classification model. A dataset containing data from 20,000 randomly generated sensors was used for the tests. The model consists of two classes, and it produces output results of either 0 or 1.

Research Design and Experimental Plan: This research was structured to assess the level of performance of the Docker-based ML model on three different platforms (GCP Cloud Run, AWS Fargate, and On-Premises). The tests on each platform were

carried out with the following steps: Docker Image Creation: The Docker image of the ML model was prepared in advance to be executed on each platform. The Docker image includes the required dependencies and the ML model itself.

GCP Cloud Run Test: A new Cloud Run service was created through the GCP Console, and the Docker image of the ML model was uploaded to the Container Registry and subsequently deployed on Cloud Run. During the test, the model's prediction performance and the service's response times were measured using 10 randomly selected data samples.

AWS Fargate Test: The created Docker image was loaded into the Elastic Container Registry and then a new Fargate container initialization configuration was created on an Elastic Container Service using the AWS Management Console and the Docker image of the ML model was initialized with this configuration. The performance evaluation was performed using the same dataset used in the GCP Cloud Run test.

On-Premises (Windows) Test: A Docker runtime environment was configured on a Windows, and the Docker image of the ML model was executed on this server. Again, performance measurements were conducted using the same dataset used in the GCP Cloud Run and AWS Fargate tests.

Selection of Measurements and Metrics: During the tests, the performance of ML model applications on different platforms was evaluated using the following metrics:

Response Time: The time it takes for an application to respond to a request from a user or a device sending the request. This is the time from the moment a request is sent to the moment it responds to the request.

Response time is a crucial indicator for evaluating the performance of a system or service. A low response time is always considered positive. High response time can negatively impact the user experience.

Memory Usage: Refers to how much system memory an application uses while running on a system or a computer. Programs and Operating systems use this memory space for temporary storage and quick access. Low memory usage improves system performance because less memory usage allows more resources to be available for other operations. Low memory usage is considered positive. High memory usage can have negative consequences for systems.

CPU Usage: Shows the total percentage of processing power consumed to run various programs. The percentage of CPU usage can vary significantly depending on what you are doing on your computer.

High CPU utilization can be considered negative and is detrimental to the user experience. Low CPU utilization is important for keeping systems running.

Network: The communication between devices for data exchange. Network status is often evaluated based on metrics such as bandwidth, latency, packet loss, etc.

Container Instance Count: A lightweight virtualization unit running in a cloud environment, encapsulating application code, runtime, system libraries, and configuration.

Request Count: The total number of requests made within a specific period of time in a system.

Request Latencies: The statistical distribution of the time interval between sending a request and receiving a response. It is often expressed with metrics such as average latency, minimum and maximum latency, etc.

Max Concurrent Request: The maximum number of requests being processed simultaneously within a given time frame.

Container Startup Latency: The time taken to start a container instance.

Thread Count: The number of threads concurrently running in a computer program. Threads are small functional units that execute concurrently in different parts of a program.

Service Count: The service count indicates how many Fargate services are currently deployed and operational in the AWS account.

Ephemeral Storage Utilization: This is temporary disk space available on a cloud virtual machine or container instance.

These performance metrics allowed for a comprehensive evaluation of the ML model's performance and cost-effectiveness across different platforms. The comparison of response time, memory usage, CPU usage, Network and more provides valuable insights into the strengths and weaknesses of each platform, aiding in the selection of the most suitable platform for deploying the ML model.

3. MATERIALS & METHOD

This section discusses the materials and methods used during the study. The criteria mentioned in Chapter 2 were undeniably important for comparing the three settings. In the following, these criteria will be discussed in more detail. As mentioned in Section 2, the ML model developed using the Python Flask library was dockerized and then deployed in three environments (AWS, GCP, Windows). The experiments were performed in the cloud for AWS Fargate and Google Cloud Run and for Windows a Windows VM was deployed using Compute Engine on Google. File uploads to GCP were supported by the Google Cloud SDK, while for AWS and Windows, direct interface uploads were performed. In the case of AWS, the dockerized file was sent to Fargate using Amazon ECR. In GCP, the Container Registry was used for this process. AWS Fargate, GCP Cloud Run and Windows naturally provide certain metrics to their users. The metrics used in this study are detailed in Section 2 and compared between Amazon Web Services, Google Cloud Platform and Windows environments. During the comparison, 1000 requests were sent to the application at rest in approximately 5 minutes and the output was recorded. The metric monitoring sections of the GCP, AWS and Windows environments helped to compare the metrics. The environments provided different metrics and the specific metric results of each environment are given in the study, of course similar 4-5 metrics were provided and compared in all 3 environments. A short script written in Python was used to measure the Response Time metric. This script shows how many seconds an application spends responding to a request.

This section provides a description of the metrics used in the study and their importance for this project. The average and maximum values of each metric are presented in the study. Response Time, Memory Utilization, CPU Utilization and Network were calculated in 3 environments. Container Instance Count was observed for AWS Fargate and Google Cloud Run. Request Count, Request Latencies, Max Concurrent Requests, Container Startup Latency could only be calculated for Google Cloud Run. Service Count and Ephemeral Storage Utilization could only be calculated for AWS Fargate. Thread Count was calculated only for Windows.

Response Time: The response times of ML model services on all three platforms were measured as the time it takes for the service to respond to a client's request. Response times were recorded in milliseconds, and a faster response time indicates better performance.

Memory Usage: The memory usage of Docker containers running on each platform was monitored. This metric was used to assess the impact of the ML model on system memory during runtime. Lower memory usage suggests better system performance.

CPU Usage: The number of CPUs utilized by the ML model was measured. This metric was employed to evaluate CPU usage and differences among the platforms. Lower CPU usage indicates a lighter workload on the system, signifying more efficient performance.

Network: Network information was provided in 3 environments used in the study. Network refers to performance measurements related to network traffic.

Container Instance Count: This metric can be observed in AWS Fargate and Google Cloud run. It shows the number of containers.

Request Count: This metric is only observable in Google Cloud Run. Number of requests reaching the service.

Request Latencies: This metric is only observable in Google Cloud Run. Distribution of request times in milliseconds reaching the service.

Max Concurrent Request: This metric is only observable in Google Cloud Run. Distribution of the maximum number number of concurrent requests being served by each container instance over a minute.

Container Startup Latency: This metric is only observable in Google Cloud Run. Distribution of time spent starting a new container instance in milliseconds.

Thread Count: This metric is only observable in Windows. The number of concurrently executing threads within a software program.

Service Count: This metric is only observable in AWS Fargate. Indicates the number of services standing to start the model.

Ephemeral Storage Utilization: This metric is only observable in AWS Fargate. It is usage of temporary storage space.

GCP provides parameters such as request count, container instance count, container CPU usage, container memory usage, and more. These metrics help you monitor the performance of your Cloud Run application, identify errors, and optimize resource usage. This way, you can make the necessary adjustments to make your service more efficient, fast and reliable. The study focused on the detailed analysis of metrics, which were common across all three environments, for performance evaluation.

Below is an example of the metrics offered by the environments.

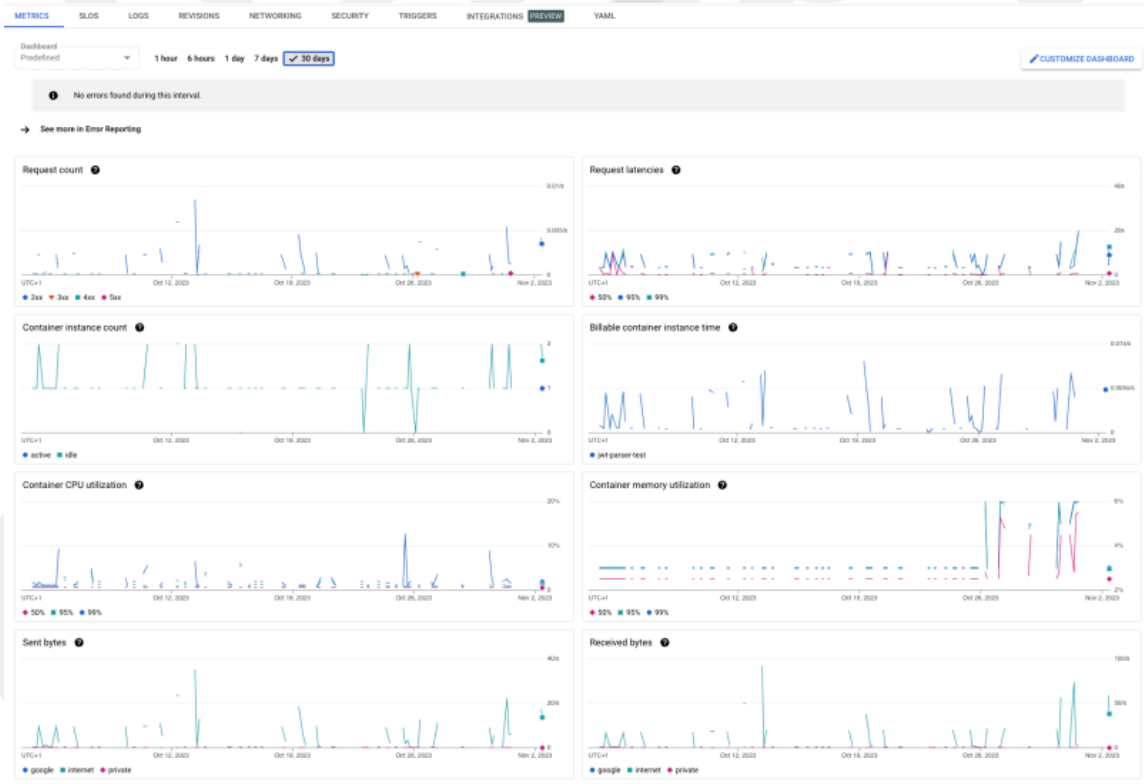

Figure 3.1: Example of GCP Metrics [33]

On the AWS side, it supports many measurement parameters like in GCP. On the AWS side, it supports many measurement parameters like in GCP. These metrics are used to monitor the performance and health of your Fargate tasks. You can collect and analyze these metrics with CloudWatch or other monitoring tools. This way, you can optimize the

performance of your applications and use resources more effectively. The metrics can be further enriched, and you can see different parameters.

Below is an examples of the AWS metrics offered by the AWS environment.

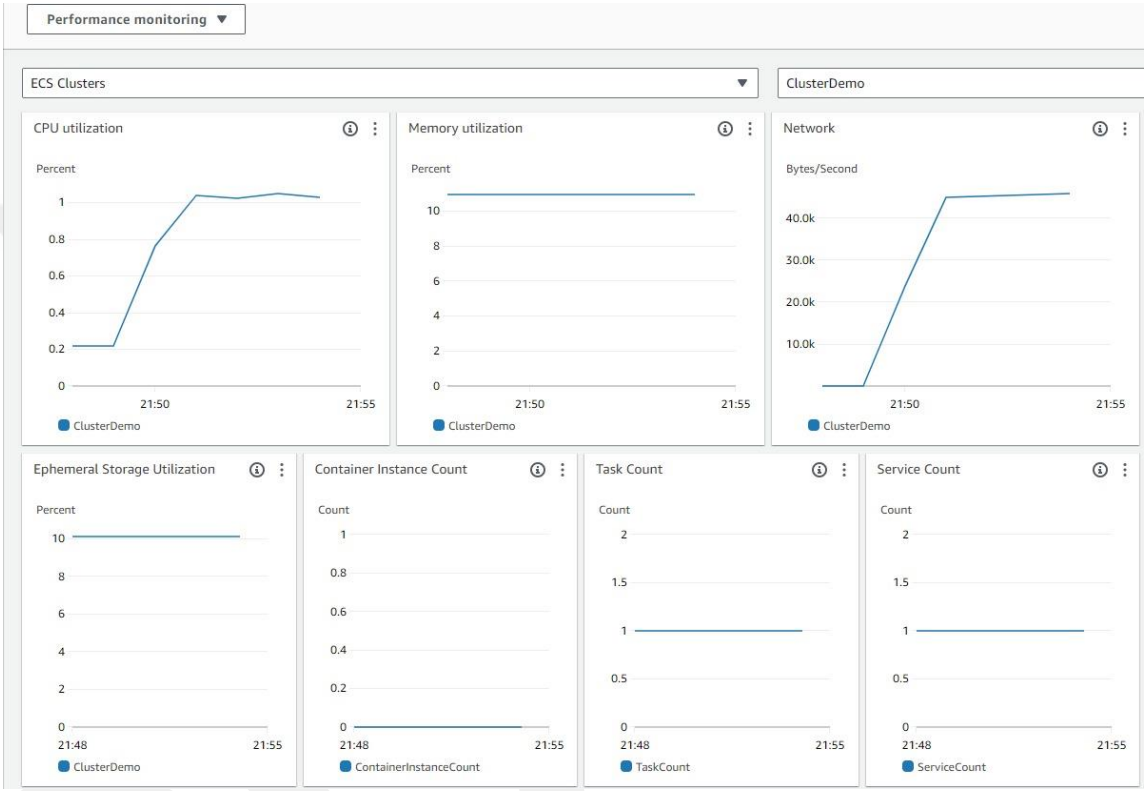

Figure 3.2: Example of AWS Metrics

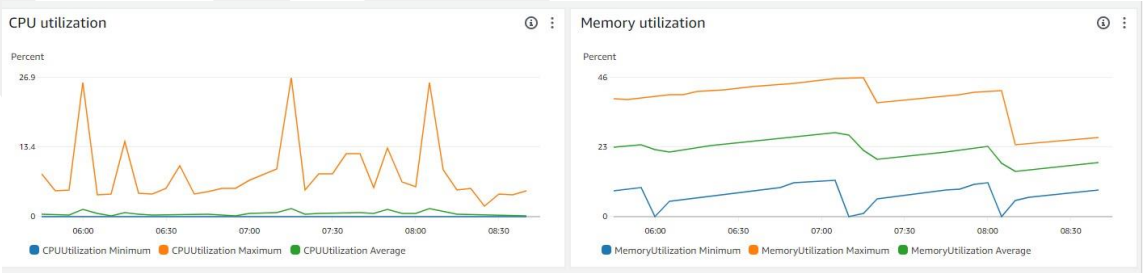

Figure 3.3: Example of AWS Metrics-2

In the Windows part of the study, different parameters were supported by Windows; those parameters are CPU utilization, Network, Memory and more. These metrics are used to monitor the performance and health of your Windows tasks. You can

collect and analyze these metrics. This way, you can optimize the performance of your applications and use resources more effectively.

Below is an example of the Windows metrics offered by the Windows environment.

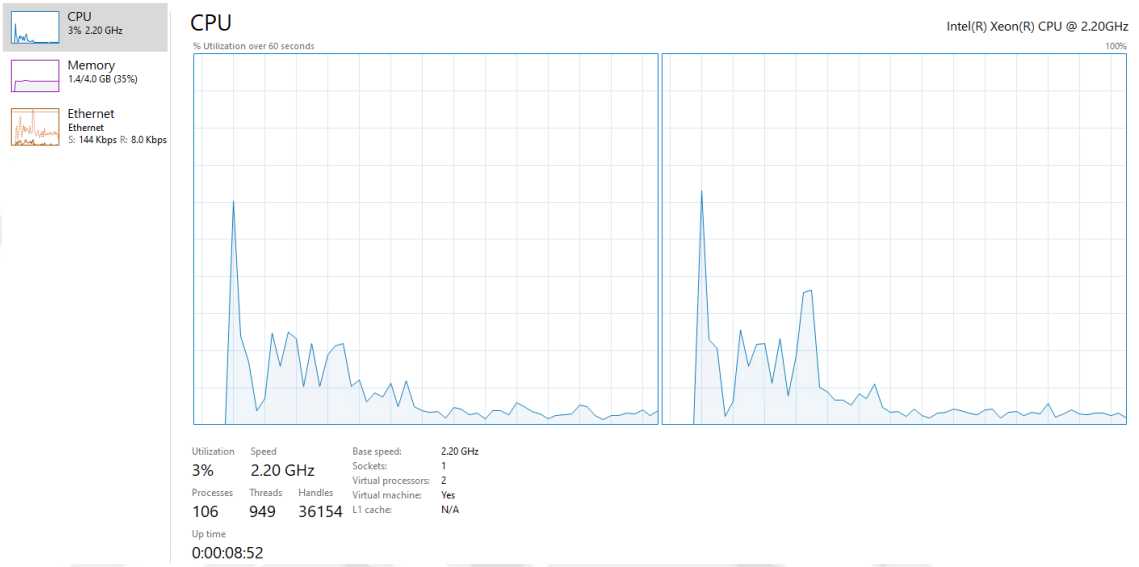

Figure 3.4: Example of Windows Metrics

Based on the metrics screens above, I calculated the metrics that explained the results in section 4 in all 3 environments. Only Python code was used to calculate the response time, using the time and request libraries to write the code. First the time when the request was sent was measured, then the time when the response was received was measured again and the response time was obtained by subtracting the two times from each other. This process was applied to each of the 3 environments separately, sending 1000 requests in 5 minutes. The result of each request was recorded and divided by the total number of requests to obtain an average response time. The results are given in section 4.

Below is the code that calculates the response time.

```python
import requests
import time

url = 'http://52.91.150.91:5000'

try:
    start_time = time.time()  # İstek gönderme zamanını kaydedelim
    response = requests.get(url)
    end_time = time.time()  # İstek sonuç alma zamanını kaydedelim

    if response.status_code == 200:
        result = response
        print("Result:")
        print(result)
    else:
        print(f"Request Error: Status Code {response.status_code}")

    # İstek yanıt süresini hesaplayalım
    response_time = end_time - start_time
    print(f"Response Time: {response_time:.3f} seconds")
except requests.exceptions.RequestException as e:
    print(f"Request Error: {e}")
```

Figure 3.5: Code that calculates Response Time

4. RESULTS

4.1 Measurement Results

This study aims to compare the performance and cost of a Machine Learning (ML) model running on Docker in GCP Cloud Run, AWS Fargate and On-Premises (Windows) environments. All environments worked in the USA region. GCP Compute Engine was used for Windows and the region was chosen as USA. Port 5000 was used in the study. AWS Fargate uses 2 vCPUs and 3 GB Memory. For Google Cloud Run, 2 vCPU and 2 GB Memory were used. For Windows, 2 vCPU and 4 GB Memory were used. Especially CPU values were chosen the same. During the measurements, applications running in the background on the Windows operating system were terminated as much as possible and their impact on the operation was set to minimum.

The findings obtained as a result of the tests and analyzes are as follows: Firstly, the response time was measured.

4.1.1 Response Time

First, response time was measured. The mean, maximum and standard deviation are given in the study. The results are as shown below, according to the results, the service with the best response time was Cloud Run, followed by AWS Fargate and then On Prem.

Table 4.1 Result of the Response Time

Platform	Response Time (Avg)	Response Time (Max)	Standard Deviation
AWS Fargate	298 Milliseconds	357 Milliseconds	0,016
GCP Cloud Run	290 Milliseconds	341 Milliseconds	0.060
Windows	319 Milliseconds	465 Milliseconds	0.017

4.1.2 Memory Utilization

The study measured memory usage and the results are shared below. According to the results, Windows was the best, followed by AWS Fargate and then Google Cloud Run.

Table 4.2 Result of the Memory Utilization

Platform	Memory Utilization (Avg)	Memory Utilization (Max)
AWS Fargate	330 MB	330 MB
GCP Cloud Run	436 MB	450 MB
Windows	300.8 MB	303 MB

4.1.3 CPU Utilization

The study measured CPU utilization and found that on average, Windows achieved minimum CPU utilization but was worst when looking at maximum CPU utilization. Google Cloud Run seems to be more consistent in average and maximum results, followed by AWS Fargate.

Table 4.3 Result of the CPU Utilization

Platform	CPU Utilization (Avg)	CPU Utilization (Max)
AWS Fargate	%0.76	%1.05
GCP Cloud Run	%0.56	%1.0
Windows	%0.30	%2.3

Total vCPU values are selected 2 vCPU all of environments so the result given as a percentage.

4.1.4 Network

The study measured the network. According to the results, Fargate is the best choice for sending data, followed by Google Cloud Run and Windows. Average and maximum values were measured.

Table 4.4 Result of the Network

Platform	Network (Avg)(Byte/Saniye)	Network (Max)(Byte/Saniye)
AWS Fargate	40000	45672
GCP Cloud Run	30993	32270
Windows	30.380	30500

4.1.5 Container Instance Count

AWS Fargate and Cloud Run provide this information, hence the values for the 2 environments are given in this measurement. AWS Fargate gave the best result using a single container.

Table 4.5 Result of the Container Instance Count

Platform	Container Instance Count (Avg)	Container Instance Count (Max)
AWS Fargate	1	1
GCP Cloud Run	1	2

4.1.6 Request Count

Only Cloud Run provides this information, so only it is included in the table.

Table 4.6 Result of the Request Count

Platform	Request Count (Avg)	Request Count (Max)
GCP Cloud Run	2.24/second	3/second

4.1.7 Request Latencies

Only Cloud Run provides this information, so only it is included in the table.

Table 4.7 Result of the Request Latencies

Platform	Request Latencies (Avg)	Request Latencies (Max)
GCP Cloud Run	5.04/ms	11.699/ms

4.1.8 Max Concurrent Requests

Only Cloud Run provides this information, so only it is included in the table.

Table 4.8 Result of the Max Concurrent Requests

Platform	Max Concurrent Requests (Avg)	Max Concurrent Requests (Max)
GCP Cloud Run	1	1.99

4.1.9 Container Startup Latency

Only Cloud Run provides this information, so only it is included in the table.

Table 4.9 Result of the Container Startup Latency

Platform	Max Concurrent Requests (Avg)	Max Concurrent Requests (Max)
GCP Cloud Run	6.51 second	7.167s

4.1.10 Thread Count

Only Windows provides this information, so only it is included in the table.

Table 4.10 Result of the Container Startup Latency

Platform	Thread Count (Avg)	Thread Count (Max)
Windows	7	12

4.1.11 Service Count

Only AWS Fargate provides this information, so only it is included in the table.

Table 4.11 Result of the Service Count

Platform	Service Count (Avg)	Service Count (Max)
AWS Fargate	1	1

4.1.12 Ephemeral Storage Utilization

Only AWS Fargate provides this information, so only it is included in the table.

Table 4.12 Result of the Ephemeral Storage Utilization

Platform	ESU (Percent)	ESU(GIB)
AWS Fargate	%10	2 GIB

DISCUSSION

The aim of this study is to show the most effective and efficient way to the user by comparing the ML model deployed on different environments with Docker based on certain criteria. As it is known, AWS, GCP as cloud, and Windows environment as On-Prem were used in this study. In this context, various performance metrics were used to understand how different platforms perform. The metrics used are the metrics supported by the environments and the results are given. These metrics are cpu usage, memory usage, network, response time and more. In the study, a detailed explanation is given for AWS, GCP and Windows and the positive and negative aspects of the services used are mentioned. In addition, the architectures used in the study are also given under separate headings. These metrics were used to determine which platform you should use, and the results are presented in section 4.

Looking at the results, Cloud Run has a faster turnaround time for Response time, with AWS Fargate in second place with an average of 0.3 seconds and Windows in third place with 0.31. This metric is important because it shows how many seconds it takes to return a response under load when your app goes live. This is why apps in the cloud have auto-scaling based on the amount of incoming load. This is why the response time metric is important. Windows did best in the memory usage results. On average and at maximum, Windows and AWS Fargate are close, but Google Cloud Run has a higher memory utilization. In CPU utilization, Windows gave the best result on average, but Windows also gave the highest result at maximum, but GCP Cloud Run and AWS Fargate gave more stable results. AWS Fargate gave the best result in network measurement in bytes and seconds. The number of container instances parameter could only be calculated by AWS Fargate and Google Cloud Run. AWS Fargate gave the best result on this parameter and used 1 container. Number of requests, Request Delays, Maximum Concurrent Requests and Container Startup Delay could be measured only in Cloud Run and only in Cloud Run. This metric is not supported in other environments. Service Count and Temporary Storage Utilization could only be calculated for AWS Fargate. Thread Count could only be calculated for Windows. In total, the best environment based on the metric varies when the result is evaluated, so you can choose the best option for you based on

which parameter is important to you. In the study, no extra different jobs were run in all 3 environments and only the docker used in the study was deployed and the results were observed. For Windows, all unnecessary and non-vital tasks running in the background were stopped and then measured.

When the studies in the literature are examined, there are studies showing performance comparisons of AWS Fargate, GCP Cloud Run services. In the literature, general comparisons of cloud providers such as AWS, GCP, AZURE are also described and comparison of these environments using different parameters is given. There are such studies in the literature, but in this study, in addition to cloud services, Windows was used and a web application was compared using different parameters such as memory usage, CPU usage, network, response time and more. Looking at similar studies, [1] concluded that Fargate is not the optimal solution for small-grained, low-memory tasks. In the study, Cloud Run has 2048MB-2vCPU and Fargate has 4096MB-2vCPU. There were no significant problems or delays in Cloud Run execution. Average execution times are given. The studies argued that Cloud Run is a better solution than Fargate but warned about the smaller resource pool and maximum execution time limit for Cloud Run. For Cloud Run, they report that it does not allow enough memory to process large tasks. The region of the study was Ireland. [2] Overall, GCP Cloud Run performed better all around. The GCP Cloud Run PaaS platform outperformed AWS EKS with Fargate CaaS platform in terms of web application performance. The results of the performance test indicate that GKE AutoPilot performed somewhat worse than both AWS EKS with AWS Fargate and GCP Cloud Run, which both performed fairly well. The sole negative aspect of GCP Cloud Run is that users could occasionally encounter a delay of more than eight seconds while submitting their first request. Price was also compared, and Cloud Run was found to be the most affordable service under certain conditions. In this study, Google Cloud Run had the highest number of parameter measurements because Google provides these parameters to the user. However, the results suggest that different services stand out in different parameters. Cloud Run gave the best result for response time, but Windows and AWS Fargate are close to each other in memory usage and CPU usage, followed by Cloud Run. In the network parameter, AWS Fargate gave the best result, followed by Cloud Run

and Windows, and more detailed results are given in section 4. Parameters such as the zone used, machine type or CPU variety may have an important role in the different results. This study has brought an innovation to the literature on Cloud vs On-Prem comparison with certain parameters. AWS, GCP and Windows are compared. The results are similar to those in the literature, but there are some differences, which may be due to the reasons I explained above.

This study used two cloud providers and one on-premises environment. The study can, of course, be expanded using different environments to obtain more robust results. For example, options such as Azure, Alibaba Cloud, and IBM Cloud can be added to the cloud side, and a Linux server can be included on the on-premises side. More complex operations can be run to test the environments, and the performance of the system can even be evaluated by sending instant requests. A more detailed study can be conducted by enriching the criterion parameters used in the study. Different parameters such as cost, security, scalability, and compliance can be considered to enrich the study. As the scope of the study expands, it is certain that it will provide ideas to more users.

If the weaknesses of the study are mentioned, of course, due to cost concerns on the cloud, these studies were carried out with attention to wages. If there were no cost concerns, a much better study could have been done. The services used in this study were deployed in only one region. It may give different results in different regions or services may give different results in different CPU, memory settings. If the look at the limits in this study, the results were analyzed over 1000 requests in 5 minutes. At this stage, a higher number of requests can be tried, or measurements can be made in seconds instead of minutes. Therefore, there is no general binding. As mentioned above, this study would be more instructive for users if it could be done using more environments, but the methods and environments used will be a roadmap for most users. More comprehensive studies can be conducted with different perspectives.

This study focused on AWS ECS-Fargate, GCP Cloud Run and on-premises deployment. It will undoubtedly help and guide future work. When similar parameters are used, close results will be obtained, of course, there are some dependencies in this study

and this should not be forgotten. In which region the services are running or CPU, memory settings are a dependency. The performance shown may vary according to all these dependencies. On the other hand, while conducting this study, cold start situations were not examined on the services, but it was observed in the study that Cloud Run cold start was observed less than AWS Fargate. Therefore, this study is preliminary information when all these conditions are met. This study is valuable because it solves an important problem and compares different environments using different parameters.

In the future, advances in technology will increase the opportunities to deliver immersive experiences through the integration of various environments and systems, enabling a wider range of research and testing of various parameters.

Multi-cloud and hybrid operating models are likely to be adopted by many companies in the future. It will enable the development of joint projects in both cloud and on-premises systems and will enable more flexible management of workloads.

Different types of machines and resources may emerge in the future, and these developments may lead to different ways of experimentation. In particular, resources such as specialized hardware accelerators, graphics processing units (GPUs) and next-generation processors could be a new milestone in terms of increased performance and scalability. It is also seen in this study and in the literature that cold start is a very time-consuming phase for services on the cloud, and if future cloud providers work on this point, services on the cloud may become more attractive.

Artificial intelligence and machine learning are entering our lives more and more every day. Cloud systems especially continue to invest in these areas. Google has made a significant investment in this area with Gemini. As these investments increase, more user-friendly and more performant services can be developed for users.

Cloud security is an important issue and companies do not look at the cloud environment very favorably due to concerns about data theft. There is no doubt that there

will be significant developments on this issue in the future and that future developments will make companies more comfortable.

All these developments will make it possible to examine more complex scenarios in future studies and to evaluate the various advantages of cloud technologies from a broader perspective.

REFERENCES

[1] K. Burkata, M. Pawlik, B. Balisa, M. Malawski, K. Vahib, M. Rynge, R. F. da Silva, E. Deelman, "Serverless containers – rising viable approach to scientific workflows," Oct. 2020, doi: 10.1109/eScience51609.2021.00014.

[2] M. F. Rahman, "Serverless Cloud Computing: A comparative analysis of performance, cost, and developer experiences in container-level services," Aug. 2023.

[3] R. Győrödi, M. I. Pavel, C. Győrödi, D. Zmaranda, "Performance of onprem versus Azure SQL Server: A case study," pp. 15894–15902, Jan. 2019, doi: 10.1109/ACCESS.2019.2893333.

[4] A. Yevge, P. Ghag, C. Solanki, A. Mishra, "Review paper on cloud service provider - AWS, Azure, GCP," Feb. 2022.

[5] Dr. M. Saraswat, Dr. R. C. Tripathi, "Cloud computing: Comparison and analysis of cloud service providers—AWS, Microsoft and Google," Dec. 2020, doi: 10.1109/SMART50582.2020.9337100.

[6] P. Wankhede, M. Talati, R. Chinchamalatpure, "Comparative study of cloud platforms Microsoft Azure, Google Cloud Platform and Amazon EC2," Apr. 2020, doi: 10.46565/jreas.2020.v05i02.004.

[7] T. Mufti, P. Mittal, B. Gupta, "A review on amazon web service (AWS), microsoft azure & google cloud platform (GCP) Services," Jan. 2021, doi: 10.4108/eai.27-2-2020.2303255.

[8] X. Cheng, A. Bounfour, "Performance analysis of public cloud computing providers" (2016). MCIS 2016 Proceedings. 25. [Online]. Available: https://aisel.aisnet.org/mcis2016/25..

[9] S. Nasrin, T. I. M. F. Sahryer, P. P. Mazumder, "Feature and performance based comparative study on serverless framework among AWS, GCP, Azure and Fissio," Dec. 2021, doi: 10.1109/ICCIT54785.2021.9689779.

[10] I. Bari, S. Babu, M. Iqbal, Dr. Y. Saleem, Z. A. Masood, "Cost and performance based comparative study of top cloud service providers," pp. 172–177, Dec. 2015.

[11] S. Kandula, M. Zhang, A. Li, X. Yang, "CloudCmp: Comparing public cloud providers," pp. 1–14, Nov. 2010, doi: 10.1145/1879141.1879143.

[12] S. Boneder, "Evaluation and comparison of the security offerings of the big three cloud service providers Amazon Web Services, Microsoft Azure and Google Cloud Platform," 2023.

[13] M. Wahlberg, "Performance comparison study of clusters on public clouds," pp. 70, 2019.

[14] M. A. Kamal, H. W. Raza, M. Alam, M. S. Mazliham, "Highlight the features of AWS, GCP and Microsoft Azure that have an impact when choosing a cloud service provider," Jan. 2020, doi: 10.35940/ijrte.D8573.018520.

[15] H. Lee, K. Satyam, G. C. Fox, "Evaluation of production serverless computing environments," July 2018, doi: 10.1109/CLOUD.2018.00062.

[16] P. Jain, Y. Munjal, J. Gera, Dr. P. Gupta, "Performance analysis of various server hosting techniques," pp. 70–77, 2020, https://doi.org/10.1016/j.procs.2020.06.010.

[17] R. V. Alvarez, L. Marino-Ramirez, D. Landsman, "Transcriptome annotation in the cloud: Complexity, best practices, and cost," Jan. 2021, doi:10.1093/gigascience/giaa163.

[18] R. Kelley, A. D. Antu, A. Kumar, B. Xie, "Choosing the right compute resources in the cloud: An analysis of the compute services offered by Amazon, Microsoft and Google," Oct. 2020, doi: 10.1109/CyberC49757.2020.00042.

[19] N. S, S. M. S, "Estimating the deployment time for containerized application using novel Microsoft Azure based docker container over Google Cloud Platform based docker container," 2022, https://doi.org/10.47750/pnr.2022.13.S04.189.

[20] L. Herrera-Izquierdo, M. Grob, "A performance evaluation between docker container and virtual machines in cloud computing architectures," Dec. 2017.

[21] J. Mart, A. Oyetoro, U. Amah, "Best practices for running workloads in public cloud environments," Apr. 2023, doi: 10.14293/PR2199.000058.v1.

[22] T. A. Bodhanya, "Comparing cloud orchestrated container platforms under the lenses of performance, cost, ease-of-use, and reliability," pp. 31, 2022.

[23] A. R. Khot, "A comparative analysis of public cloud platforms and introduction of multi-cloud," Sept. 2020, doi: 10.38124/IJISRT20SEP234.

[24] A. Abraham, J. Yang, "A comparative analysis of performance and usability on serverless and server-based Google Cloud Services," pp. 408–422, May 2023, doi: 10.1007/978-3-031-33743-7_33.

[25] A. Goli, O. Hajihassani, H. Khazaei, O. Ardakanian, M. Rashidi, T. Dauphinee, "Migrating from monolithic to serverless: A fintech case study," pp. 20–25, Apr. 2020, https://doi.org/10.1145/3375555.3384380.

[26] J. M. Menendez, J. E. L. Gayo, E. R. Canal, A. E. Fernandez, "A comparison between traditional and serverless technologies in a microservices setting," May 2023 https://doi.org/10.48550/arXiv.2305.13933..

[27] I. Goswami, "Serverless architecture for machine learning" (2023). Master's Projects. 1336. https://scholarworks.sjsu.edu/etd_projects/1336.

[28] D. Chahal, R. Ojha, M. Ramesh, R. Singhal, "Migrating large deep learning models to serverless architecture," 2020, doi: 10.1109/ISSREW51248.2020.00047.

[29] O. Paakkunainen, "Serverless computing and faas platform as a web application backend," Jun. 2019.

[30] A. Chauhan, "A comparative study of cloud computing platforms," pp. 821–826, Apr. 2020, https://doi.org/10.17762/turcomat.v11i1.13563.

[31] [https://coskunkurtuldu.medium.com/bulut-bilisim-nedir-azure-69767979d90], (Date Last Accessed: 24 Oct 2023).

[32] [https://www.goldfinchgrp.com/post/ml-system-in-production], (Date Last Accessed: 24 Oct 2023).

[33] [https://www.marcinkwiatkowski.com/cloud-2/unlocking-the-power-of-google-cloud-run-in-2023/], (Date Last Accessed: 02 Nov 2023).

[34] [https://ekremkurt1907.medium.com/saas-paas-iaas-9e69db47f005], (Date Last Accessed: 24 Aug 2023).