Evaluating Execution Times and Costs of a Federated Learning Application on different Cloud Providers

Rafaela C. Brum[1,2], Luciana Arantes[2], Maria Clicia Castro[3], Pierre Sens[2], and Lúcia M. A. Drummond[1]

[1]Fluminense Federal University - Niterói, Brazil
[2]Sorbonne Université, CNRS, INRIA, LIP6 - Paris, France
[3]State University of Rio de Janeiro – Rio de Janeiro, Brazil

Résumé
Federated Learning (FL) is a new area of distributed Machine Learning (ML) that emerged to deal with data privacy concerns. In FL, each client has access to a local private dataset. At every round, a client trains the model with its local dataset and sends the weights to a central server. The latter aggregates all client weights and then sends the final weights back to the clients. This approach is attractive in many domains as it allows multiple institutions to collaborate on an ML task without sharing their data. However, most ML models used in FL have millions of weights exchanged in each message. The messages sent between a client and the server can achieve gigabytes of size and are exchanged several times in the whole FL execution. This work presents a preliminary analysis of execution times and costs of a FL application in a multi-cloud scenario. Experiments were conducted considering executions on the Amazon Web Services, Google Cloud Provider, and also in both cloud providers at the same time.

Mots-clés : Cloud Computing, Federated Learning, Time and Cost Evaluation

1. Introduction

Federated Learning (FL) is a recent type of distributed Machine Learning (ML) in which the participating clients do not share their private data [29]. The clients federation solves the learning task coordinated by a central server without sharing the data. Instead, each client computes and communicates only the model weights to update the current global model kept by the server. This server-clients architecture of FL is classified into Cross-Device or Cross-Silo Federated Learning, depending on the connected client's type. A Cross-Device Federated Learning has low-powered devices as clients [19, 21] while a Cross-Silo Federated Learning has different companies or institutions as clients (*e.g.*, hospitals [22]) with private datasets willing to collaborate to create a global model. In this work, we focus on the second type of Federated Learning, in which the central server can assume that all clients are available during the whole execution, and there are usually fewer clients (less than 10 [15]).
McMahan *et al.* [19] proposed the term Federated Learning in 2017 as a learning technique that allows users to collectively benefit from shared models trained from distributed data without centrally storing them. The authors presented a Cross-Device FL in which the clients are mobile phones. Since then, this area has received much attention from researchers due to the increasing concern with data privacy. In traditional distributed ML, participants usually exchange data to

balance their execution times. However, with the rise of data protection laws (*e.g.*, GPDR [1] in Europe and LGPD [2] in Brazil), application domains that handle sensitive data cannot share it anymore and depends on FL to create good prediction models.

Moreover, the digital data created by an institution and used to train ML algorithms increases rapidly. Most institutions cannot upgrade their data centers due to high financial costs. One viable option is the use of cloud storage services, in which the user only pays for the amount of stored data. Many cloud providers offer these storage services with different privacy guarantees and data availability. For example, in the Google Cloud provider, the user can store the data in a multi-region scenario. The data is available in different regions of the same country, and the user can determine who can access it.

In this scenario, each participating institution of the FL environment can choose the best storage option among the different cloud providers. According to Li *et al.* [14], the response time to access a file in one cloud provider can be twice the time to access it in another provider. Besides, they show that the maximum throughput of one cloud provider can vary up to 57%, and in another cloud provider varies less than 2%, depending on the cloud region used. Moreover, inside the same cloud provider, diverse storage options have different response times and throughput, as presented by Teylo *et al.* in [31]. Thus, each institution chooses the best cloud provider, cloud region, and storage service combination based on its respective needs, leading to a scenario where FL clients have their data in different cloud providers and, to preserve privacy, it is prohibitive to transfer all data to the same cloud provider.

In previous works [2, 3], we present FL scenarios for a biomedical application that searches for tumor-infiltrated lymphocytes to estimate the patient survival rate using Convolutional Neural Networks (CNNs) as models. These models extract knowledge from images using millions of neurons, which are grouped into layers. Each neuron consists of an activation function and receives the results of the previous layer multiplied by weights, which are arbitrary float pointing values, as input. Thus, to train the model, it is necessary to multiply each image (represented by a matrix) by all layers of the network several times, which requires a huge computational power. One option to achieve this performance is using accelerators, such as Graphics Processing Units (GPUs).

Besides the storage services, cloud providers offer Virtual Machines (VMs) with GPUs in a service generically called Infrastructure-as-a-Service (IaaS). The user only pays for what he/she uses (pay-as-you-go scheme), with no costs for hardware maintenance or energy consumption. Furthermore, the user has access to the newest computational resources as soon as the provider renders them available to hire. In Amazon Web Services (AWS), for example, there are many predefined types of VMs, also called instances, with different NVIDIA GPU architectures, from Kepler to Volta [27]. In Google Cloud Platform (GCP), the user can attach GPUs to a predefined or custom instance type [11].

To the best of our knowledge, few works tackle the problem of FL on clouds in the related literature. Liu *et al.* [18] present a hierarchical FL architecture with mobile devices as clients while Fang *et al.* [5] propose an architecture for Federated Learning in the cloud focusing on privacy. However, both of them simulate the FL environment in a local machine and present only these results in their evaluation. Rajendran *et al.*present in [22] a FL approach whose results concern two Machine Learning models in a scenario with two clients in both a simulated environment and on Microsoft Azure Cloud Databricks [4], an open-source tool for data engineering and collaborative data science, to exchange the ML model between two institutions.

1. https ://gdpr-info.eu/
2. http ://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/L13709compilado.htm

In our previous work, all the FL clients were in the same zone of Amazon Web Services (AWS), in *us-east-1* (North Virginia). As explained before, this scenario does not properly reflect real choices, as institutions can store their datasets in different cloud providers and cloud regions. Thus, in this paper, we investigate how the location of both clients and datasets impacts the total FL execution time, and particularly the spent time in the synchronization steps in different cloud scenarios. Thus, we deploy the clients and the server across two providers, AWS and GCP. Our results show that the allocation of clients and servers has a key role in the execution times and financial costs of the FL application.

The remaining of this paper is organized as follows. Section 2 presents concepts of cloud infrastructures. The used application in our experiments, a Federated Learning for solving a Tumor-Infiltrating Lymphocytes Classification Problem, is presented in Section 3. Section 4 shows our experimental results. Finally, Section 5 presents some conclusions and future work.

2. Cloud Infrastructures

Cloud providers divide their physical infrastructure into cloud regions, which are independent and isolated geographic areas [28, 9]. For example, the first region in the eastern United States in AWS is in North Virginia (*us-east-1* region), while in GCP, it is in South Carolina (*us-east1* region). We can see a map with all cloud regions in AWS' or on GCP's website [26, 10].

The price of a VM type in each cloud provider varies among different cloud regions. For example, in April 2022, the GCP's VM type *e2-standard-4*, with 4 vCPUs and 16 GB of RAM, costed \$0.13402 in Iowa (*us-central1* region) and \$0.16098 in Salt Lake City (*us-west3* region) [3]. While the AWS' VM type *t2.xlarge*, with the same configuration, costed \$0.1856 in North Virginia (*us-east-1* region) and \$0.2208 in North Carolina (*us-west-1* region).

Besides the VM allocation costs, there are two other costs to be considered : (i) the communication costs, since an FL application exchanges many messages during its execution, and (ii) the storage systems costs, necessary to store the whole dataset. In each FL round, the server sends all model weights to each client, receives them back after the training, and sends the aggregated ones so that the clients can evaluate the model. In terms of size, the VGG16 model, used in our previous work [3], has 132 million weights, which are float pointing variables using a total of 5.2 GB of memory. Thus, in each FL round, the server exchanges up to 15.6 GB in messages. In AWS, the price is \$0.09 per GB sent from AWS to the Internet in the first 10 TB/month. In GCP, there are two different network tiers, called Standard and Premium Tier [12]. While the Premium Tier uses GCP's high-speed internal network as much as possible, the Standard Tier uses the Internet to deliver the exchanged data. In the Premium Tier, we pay \$0.12 per GB sent from GCP to any destination worldwide, except in China and Australia, in the first 1 TB/month, after this first TB, we pay \$0.11 per GB. In the Standard Tier, we pay \$0.085 per GB sent from GCP to any destination worldwide, except in China and Australia, in the first 10 TB/month.

Regarding the storage services, AWS offers a total of 11 storage services [25] and GCP offers nine different storage services [8]. Each storage service focuses on different needs in a company's workflow. In this work, we use the object storage service from each of the providers : Amazon Simple Storage Service (Amazon S3) [24] and Cloud storage [6]. These services similarly represent the objects, using a two-level organization [24, 6]. At the higher level, they use buckets, structures similar to folders having a unique global name. These buckets help to organize the data of different users, identifying and billing them accordingly. S3 restricts each bucket to a single region, and each account can associate up to 100 buckets with it. In Cloud

3. https ://cloud.google.com/compute/vm-instance-pricing#n1_predefined

Storage, the user can configure the bucket availability to a single cloud region, in two close regions (dual-region), or several regions spread in a larger area (multi-region). There is no limit on the number of buckets per account in GCP, but there are limitations regarding the bucket's name and creation rate [7].
Objects are the lower level of these two storage services. They contain the user stored data represented by a name and a unique key used to access the object [4, 5]. Both services have an upper limit to a single object size of 5TB [24, 7] and allow the user to create, change and, read objects from a bucket using a single operation. However, if the user wants to rename or move the object to another place, it takes at least two operations, downloading the object to a local system and uploading it with the new name or to the new location.
The two cloud providers allow users to choose the privacy level for each object. By default, AWS makes all objects stored in S3 private, allowing only access from the resource owner and account administrator [6]. If a user wants to let others see his/her data, he/she needs to grant access to each object explicitly. On the other hand, GCP does not assume any privacy level but requests the user to set the requested level of external access when uploading new files to Cloud Storage [7].
AWS and GCP charge for storing the data and for each operation in it, and the price varies among different regions. In Amazon S3, users pay $0.023 per GB to store their data in N. Virginia (*us-east-1* region) and $0.005 per 1000 operations in this region while in N. Carolina (*us-west-1* region), users pay $0.026 per GB stored and $0.0055 per 1000 performed operations. In Google Cloud Storage, users also pay $0.023 per GB to store their data in N. Virginia (*us-east4* region) and $0.020 per GB in Iowa (*us-central1* region). The price per operation is the same in all regions of Cloud Storage, being $0.005 per 1000 operations.

3. Federated Learning for solving a Tumor-Infiltrating Lymphocytes Classification Problem

In this work, we have executed the FL approach proposed in [3] to a biomedical application, which deals with a Tumor-Infiltrating Lymphocytes (TIL) classification problem, described in [23], in different cloud scenarios. This application receives as input Whole-Slide Images (WSIs) and presents TIL maps as results. These maps show the spatial distribution and the density of TILs in each patient to help cancer treatment. There are two main phases in this application : training and production. In the training phase, the application divides each WSI into patches of smaller sizes, presents them to experts to classify as TIL-positive or TIL-negative, and uses them as a training dataset for the CNN. In the production phase, new WSIs are divided into patches and the CNN classifies these new patches into having or not having TILs. With the TIL-positive patches, the application creates the TIL maps.
The FL approach focuses on the CNN training part and implements the VGG16 model [30]. The CNN input size is 224×224, the patch dimension from the previous steps. We set the number of output classes to two, positive if the patch contains a TIL or negative otherwise. The centralized approach needs two separated datasets to train the CNN : one to train the model and another to test it. In our FL approach, we divided the two datasets among all clients homogeneously. Thus, each FL client accesses two separated and private datasets : one for training the local model and another to test it.
The FL approach is composed of communication rounds which in turn are composed of 5 steps.

4. https ://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingObjects.html
5. https ://cloud.google.com/storage/docs/naming-objects
6. https ://aws.amazon.com/s3/security/ ?pg=ln&sec=be#Access_management_and_security
7. https ://cloud.google.com/storage/docs/access-control

In the first step, the server sends the current model weights to all participating clients. After receiving the weights, each client trains for several epochs on the local training dataset and sends the updates back to the server (step 2). Then, the server receives all updates, aggregates them, and sends the final weights to the clients (step 3). Next, each client updates their weights, tests the model with the testing dataset, and sends its evaluation metrics (*e.g.*, accuracy and precision) to the server (step 4). Finally, the server aggregates the evaluation metrics of all clients to present global metrics (step 5).

4. Experimental Results

Our experiments were conducted on the framework Flower, a FL framework proposed by Beutel *et al.* [1] that focuses on the FL execution in real scenarios. The authors developed Flower to support any ML framework underneath it (TensorFlow, PyTorch, or a custom one) and several client environments, with different operating systems or hardware settings. Moreover, they implemented some FL algorithms, like FedAvg [19], FedProx [16], and Q-FedAvg [17], and the communication layer between server and clients using the gRPC protocol [13], a high-performance implementation of the Remote Procedure Call (RPC) protocol that provides communication among several tasks with minimum overhead. Since Flower is open-sourced, it is possible to modify the source code to add specific log messages or create other FL algorithms within it. We used the Python 3.6 programming language and executed both server and clients into different VMs of AWS and GCP. We deployed each task into a different VM to avoid unintended data access from other clients and reinforce the privacy issues.

Regarding the application execution parameters, we executed the FL application with 4 clients and 10 communication rounds using 5 training local epochs in each, which are the best execution parameters, as presented in [3]. Each client works on a dataset containing 948 training patches and 522 test patches collected from the TCGA repository [20]. We added log messages into the FL framework to get the time the server and clients send and receive each message. From these log messages, we computed the FL synchronization time and the computation time of each client. The FL synchronization time is the sum of three times per round : (1) the initial sync time, (2) the aggregation sync time, and (3) the test sync time. The initial sync time is the time between the server sending the first message with the current weights and the last client receiving it. The aggregation sync time is the time between the first client sending the updated weights back to the server and the last client receiving the final weights, which includes the server aggregation in step 3. The test sync time is the time between the first client sending the evaluation metrics and the server presenting the global metrics. The computation time of each client is the sum of the training and the testing steps in each client, including the time to access the dataset. We present here only the longest computation time.

Concerning the cloud environments, in both providers, each client executes in a VM with 8 vCPUs, 32GB of memory, and an Nvidia T4 Tensor Core GPU while the server in a VM with 4 vCPUs and 16GB of memory, as the server does not need a GPU. The cloud region, storage service, type of VMs (client and server), VM price (client and server), and network costs are summarized in Table 1 for the two cloud providers. We chose the Premium Network Tier of GCP to use the high-speed internal network as much as possible.

4.1. Execution times and Financial Costs

Table 2 presents the averages of three executions in different scenarios, where the costs are presented in $, and the time in hours, minutes and seconds. Our initial tests aimed at evaluating the execution times and financial costs when the complete FL application was executed on a

TABLE 1 – Cloud region, storage service, VM and cost for both client and server, and network transfer costs in each cloud providers used in our experiments

	Region	Storage Service	Client		Server		Network Costs ($ per GB)
			VM	Cost ($ per hour)	VM	Cost ($ per hour)	
AWS	N. Virginia (us-east-1)	Amazon S3	g4dn.2xlarge	0.752	t2.xlarge	0.1856	0.090
GCP	Iowa (us-central1)	Cloud Storage	n1-standard-8	0.730	e2-standard-4	0.13402	0.120

unique cloud provider. As can be seen in S2, GCP presented better results than the AWS ones (seen in S1). Even if more expensive than the standard one, the choice of the Premium network showed to be a good option when compared with the AWS results. The second set of tests evaluated two cases : (i) the server, three clients, and their corresponding data sets on GCP, and one client and the corresponding data set on AWS, case S3 ; and (ii) all clients and the server on GCP, but one data set allocated on AWS, case S4. This last test showed that the allocation of the client in the same cloud provider of the dataset is the best option, in this case, even considering that GCP has presented better results than AWS in the other cases. Finally, we considered two scenarios where two datasets and two clients were allocated to each cloud provider. In the first case, the server was allocated on AWS, and in the second case on GCP. We observed similar execution times in the two cases, despite the use of the Premium Tier of GCP, with a 12% increase in the total costs using the server in GCP.

TABLE 2 – Average times and costs of 4 scenarios : S1- all FL application on AWS, S2- all FL application on GCP, S3 - 3 clients and 3 data sets on GCP, 1 client and 1 data set on AWS, S4- 4 clients and 3 data sets on GCP and one data set on AWS.

Scenarios	**Total time**	**Total cost $**	**Syncronization time**	**Message exchange cost $**	**Computing time**	**VM cost $**
S1 (AWS)	1 :28 :32	10.51	0 :34 :47	5.80	1 :17 :55	4.71
S2 (GCP)	0 :36 :54	9.61	0 :04 :35	7.73	0 :29 :44	1.88
S3 (3 GCP,1 AWS)	1 :25 :22	12.04	0 :56 :58	7.57	1 :15 :42	4.47
S4 (4 GCP,1 DS AWS)	1 :48 :21	13.25	1 :18 :26	7.73	1 :45 :20	5.52

5. Conclusion

This paper presented a preliminary analysis of FL executions in several scenarios of clients and server allocation on GCP and AWS clouds. We could observe different costs and execution times when using those providers, even using similar infrastructures. These experiments showed that the allocation problem of clients and servers is not trivial and that the financial costs and execution times can vary in accordance with the used infrastructure. Moreover, it is not worth using the best VMs or network in one cloud provider if we have a dataset allocated in another one with worse performance, since the bigger access time of the slowest cloud provider will dominate the execution time. In future work, we intend to model that problem using optimization tools in order to obtain, given the characteristics and location of the datasets, and cloud providers' costs and infrastructure performance, the best clients and server allocation with the aim of reducing execution times and/or financial costs.

Acknowledgment

This research is supported by project CNPq/AWS 440014/2020-4, Brazil, by *Programa Institucional de Internacionalização* (PrInt) from CAPES (process number 88887.310261/2018-00) and CNPq (process number 145088/2019-7).

Bibliographie

1. Beutel (D. J.), Topal (T.), Mathur (A.), Qiu (X.), Parcollet (T.) et Lane (N.). – Flower : A friendly federated learning research framework. *ArXiv*, vol. abs/2007.14390, 2020.
2. Brum (R.), Drummond (L.), Castro (M. C.) et Teodoro (G.). – Towards optimizing computational costs of federated learning in clouds. – In *2021 International Symposium on Computer Architecture and High Performance Computing Workshops (SBAC-PADW)*, pp. 35–40, 2021.
3. Brum (R.), Teodoro (G.), Drummond (L.), Arantes (L.), Castro (M.) et Sens (P.). – Evaluating federated learning scenarios in a tumor classification application. – In *Anais da VII Escola Regional de Alto Desempenho do Rio de Janeiro*, pp. 6–10, Porto Alegre, RS, Brasil, 2021. SBC.
4. Docs (M.). – What is Azure Databricks ? – `https://docs.microsoft.com/en-us/azure/databricks/scenarios/what-is-azure-databricks`, 2022. Accessed 16 January 2022.
5. Fang (C.), Guo (Y.), Wang (N.) et Ju (A.). – Highly efficient federated learning with strong privacy preservation in cloud computing. *Computers & Security*, vol. 96, 2020, p. 101889.
6. Google Cloud (P.). – Cloud Storage. – `https://cloud.google.com/storage`, 2021. Accessed 19 December 2021.
7. Google Cloud (P.). – Quotas & limits - Cloud Storage. – `https://cloud.google.com/storage/quotas`, 2021. Accessed 19 December 2021.
8. Google Cloud (P.). – Cloud Computing Services. – `https://cloud.google.com/products/storage`, 2022. Accessed 14 April 2022.
9. Google Cloud (P.). – Geography and regions - Documentation. – `https://cloud.google.com/about/locations`, 2022. Accessed 15 February 2022.
10. Google Cloud (P.). – Global Locations - Regions & Zones. – `https://cloud.google.com/about/locations`, 2022. Accessed 15 February 2022.
11. Google Cloud (P.). – GPUs on Compute Engine. – `https://cloud.google.com/compute/docs/gpus`, 2022. Accessed 19 January 2022.
12. Google Cloud (P.). – Network Service Tiers overview. – `https://cloud.google.com/network-tiers/docs/overview`, 2022. Accessed 12 April 2022.
13. gRPC Authors. – gRPC. – `https://grpc.io/`, 2022. Accessed 04 January 2022.
14. Li (A.), Yang (X.), Kandula (S.) et Zhang (M.). – CloudCmp : Comparing Public Cloud Providers. – In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, IMC '10*, IMC '10, p. 1–14, New York, NY, USA, 2010. Association for Computing Machinery.
15. Li (Q.), Diao (Y.), Chen (Q.) et He (B.). – Federated learning on non-iid data silos : An experimental study. *ArXiv*, vol. abs/2102.02079, 2021.
16. Li (T.), Sahu (A. K.), Zaheer (M.), Sanjabi (M.), Talwalkar (A.) et Smith (V.). – Federated optimization in heterogeneous networks. *arXiv preprint arXiv :1812.06127*, 2018.
17. Li (T.), Sanjabi (M.), Beirami (A.) et Smith (V.). – Fair resource allocation in federated learning, 2020.
18. Liu (L.), Zhang (J.), Song (S.) et Letaief (K. B.). – Client-edge-cloud hierarchical federated learning. – In *2020-2020 IEEE International Conference on Communications*, 2020.
19. McMahan (B.), Moore (E.), Ramage (D.), Hampson (S.) et y Arcas (B. A.). – Communication-

Efficient Learning of Deep Networks from Decentralized Data. – In Singh (A.) et Zhu (J.) (édité par), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, *Proceedings of Machine Learning Research*, volume 54, pp. 1273–1282, Fort Lauderdale, FL, USA, 4 2017. PMLR.

20. National Human Genome Research Institute. – The Cancer Genome Atlas. – https ://cancergenome.nih.gov/, June 2017.
21. Nguyen (V.-D.), Sharma (S. K.), Vu (T. X.), Chatzinotas (S.) et Ottersten (B.). – Efficient federated learning algorithm for resource allocation in wireless iot networks. *IEEE Internet of Things Journal*, vol. 8, n5, 2021, pp. 3394–3409.
22. Rajendran (S.), Obeid (J. S.), Binol (H.), D'Agostino (R.), Foley (K.), Zhang (W.), Austin (P.), Brakefield (J.), Gurcan (M. N.) et Topaloglu (U.). – Cloud-Based Federated Learning Implementation Across Medical Centers. *JCO Clinical Cancer Informatics*, no5, 2021, pp. 1–11. – PMID : 33411624.
23. Saltz (J.), Gupta (R.), Hou (L.), Kurc (T.), Singh (P.), Nguyen (V.), Samaras (D.), Shroyer (K. R.), Zhao (T.), Batiste (R.) et al. – Spatial Organization And Molecular Correlation Of Tumor-Infiltrating Lymphocytes Using Deep Learning On Pathology Images. *Cell reports*, vol. 23, n1, 2018, p. 181.
24. Services (A. W.). – Amazon S3. – `https://aws.amazon.com/s3/`, 2021. Accessed 19 December 2021.
25. Services (A. W.). – Cloud Storage on AWS. – `https://aws.amazon.com/products/storage/`, 2022. Accessed 12 April 2022.
26. Services (A. W.). – Global Infrastructure. – `https://aws.amazon.com/about-aws/global-infrastructure/`, 2022. Accessed 15 February 2022.
27. Services (A. W.). – Recommended GPU Instances. – `https://docs.aws.amazon.com/dlami/latest/devguide/gpu.html`, 2022. Accessed 19 January 2022.
28. Services (A. W.). – Region and Zones - Amazon Elastic Compute Cloud. – `https://aws.amazon.com/about-aws/global-infrastructure/`, 2022. Accessed 15 February 2022.
29. Shen (S.), Zhu (T.), Wu (D.), Wang (W.) et Zhou (W.). – From distributed machine learning to federated learning : In the view of data privacy and security. *Concurrency and Computation : Practice and Experience*, vol. n/a, nn/a, 2020.
30. Simonyan (K.) et Zisserman (A.). – Very deep convolutional networks for large-scale image recognition. – In *3rd Int. Conf. on Learning Representations, ICLR 2015*, 2015.
31. Teylo (L.), Brum (R. C.), Arantes (L.), Sens (P.) et Drummond (L. M. d. A.). – Developing Checkpointing and Recovery Procedures with the Storage Services of Amazon Web Services. – In *49th International Conference on Parallel Processing - ICPP : Workshops*, *ICPP Workshops '20*, ICPP Workshops '20, New York, NY, USA, 2020. Association for Computing Machinery.