Received 2 August 2022, accepted 2 September 2022, date of publication 12 September 2022, date of current version 26 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3206366

TOPICAL REVIEW

Serverless on Machine Learning: A Systematic Mapping Study

AMINE BARRAK, FABIO PETRILLO, AND FEHMI JAAFAR
Département d'informatique et mathématique, Université du Québec à Chicoutimi, Chicoutimi, QC G7H 2B1, Canada
Département de génie logiciel et des technologies de l'information, École de Technologie Supérieure, Montreal, QC H3C 1K3, Canada

Corresponding authors: Amine Barrak (mabarrak@uqac.ca), Fabio Petrillo (fabio.petrillo@etsmtl.ca), and Fehmi Jaafar (fehmi_jaafar@uqac.ca)

ABSTRACT Machine Learning Operations (MLOps) is an approach to managing the entire lifecycle of a machine learning model. It has evolved over the last years and has started attracting many people in research and businesses in the industry. It supports the development of machine learning (ML) pipelines typical in the phases of data collection, data pre-processing, building datasets, model training, hyper-parameters refinement, testing, and deployment to production. This complex pipeline workflow is a tedious process of iterative experimentation. Moreover, cloud computing services provide advanced features for managing ML stages and deploying them efficiently to production. Specifically, serverless computing has been applied in different stages of the machine learning pipeline. However, to the best of our knowledge, it is missing to know the serverless suitability and benefits it can provide to the ML pipeline. In this paper, we provide a *systematic mapping study* of machine learning systems applied on serverless architecture that include 53 relevant studies. During this study, we focused on (1) exploring the evolution trend and the main venues; (2) determining the researchers' focus and interest in using serverless on machine learning; (3) discussing solutions that serverless computing provides to machine learning. Our results show that serverless usage is growing, and several venues are interested in the topic. In addition, we found that the most widely used serverless provider is AWS Lambda, where the primary application was used in the deployment of the ML model. Additionally, several challenges were explored, such as reducing cost, resource scalability, and reducing latency. We also discuss the potential challenges of adopting ML on serverless, such as respecting service level agreement, the cold start problem, security, and privacy. Finally, our contribution provides foundations for future research and applications that involve machine learning in serverless computing.

INDEX TERMS Serverless, FaaS, function as a service, machine learning, systematic mapping, systematic literature review, SM, SLR.

I. INTRODUCTION

Cloud computing is beneficial to businesses of all sizes in the marketing sector. It offers the abstraction of online services hosted on the cloud rather than complex local infrastructure. These services include everything from simple cloud storage to cloud infrastructure platforms. Cloud computing offers different benefits *i.e.,*high speed, efficiency and cost reduction, data security, scalability, back-up and data restore, control and level access, and unlimited storage capacity [1]. These services are offered in different proportions according to the provided service, such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS), Function as a Service (FaaS) or Serverless.

Moreover, these platforms have grown significantly over the last decade and are widely adopted for the delivery of computing services. In particular, serverless computing provides a simplified architecture in which code execution is fully managed by the cloud provider , in such case, developers can focus only on code writing, increasing their productivity. Recently, serverless has been used as an infrastructure to build total ML pipelines or partially with faster deployment and elastic scalability.

The associate editor coordinating the review of this manuscript and approving it for publication was Yongming Li.

On the one hand, serverless popularity is increasing, and it is receiving attention from developers, especially after Amazon launched AWS Lambda in November 2014.[1] Recently, Wen *et al.* [2] found that questions about Serverless on StackOverflow have grown 380% from 2015 to 2020. The size of the serverless market is estimated to grow from 3.33 USD Billion in 2018 to USD 31.53 Billion in 2026 [3]. On the other hand, ML has been widely used in cloud computing, mainly when it is divided into a small pipeline stage (ML as a Service). The need becomes to use Serverless since the high cost of cloud resource management.

Thus, Serverless computing [4] is an interesting option regarding the resolution of small tasks, mainly when companies cannot estimate the traffic of their ML applications, scalability, and cost accurately [5]. Furthermore, several studies are exploiting serverless computing to accomplish tasks of the ML pipeline, such as training [6], hyperparameter tuning [7], and model deployment [8].

This paper aims to map the current state-of-the-art to understand how Serverless was used in the machine learning pipeline and the challenges and opportunities for different stakeholders.

For achieving this **goal**, we perform a systematic mapping to answer three research questions by analyzing relevant studies. First, our study identified, classified, and evaluated the current state-of-the-art in machine learning on Serverless architecture. Next, we selected 50 primary studies from the Scopus database; then, we rigorously classified the studies to precisely categorize research results on ML and Serverless challenges.

The **audience** of this study is composed of both (i) researchers interested in contributing to this research area and (ii) practitioners interested in understanding existing research on machine learning applying Serverless architecture.

The main **contributions** of this study is to respond these research questions:

- WHAT ARE THE **PUBLICATION TRENDS** OF RESEARCH STUDIES ABOUT SERVERLESS ON MACHINE LEARNING?
- WHAT IS THE **FOCUS OF RESEARCH** OF APPLIED MACHINE LEARNING ON SERVERLESS COMPUTING ?
- WHAT ARE THE **POTENTIAL CHALLENGES OF ADOPTING** MACHINE LEARNING ON SERVERLESS COMPUTING?

The rest of the paper is organized as follows. In Section II we set the stage by giving the basic concepts around machine learning lifecycle and the serverless architecture. The design of the study is presented in Section III, whereas its results are elaborated in Sections IV. We have made a discussion in Section IV-C where we broadened our perspective and the potential implications for both researchers and practitioners. Threats to validity and related work are described in Sections V and VI. With Section VII, we close the paper and discuss future work.

[1]https://docs.aws.amazon.com/lambda/latest/dg/lambda-releases.html

II. BACKGROUND

This section provides background information on defining Serverless and ML pipeline, as we found during our systematic mapping.

A. MACHINE LEARNING PIPELINE

Microsoft team members introduced a typical ML pipeline [10], where they show a series of steps chained together to form the machine learning workflow essential stages. These stages include data and model-oriented artifacts from data collection and cleaning until model evaluation deployment. These stages construct an ML pipeline lifecycle. Recently, with the commercial use of AI, the MLOps field has been introduced, aiming to automate the ML pipeline [11]. A standard ML pipeline broadly consists of the following five stages shown in Figure 1:

- *Data retrieval:* is the process of identifying and extracting data from a database, based on a query provided by the user or application.
- *Data preparation:* is the process of gathering, combining, structuring and organizing data.
- *Model training:* The process of training an ML model involves providing the data features to an ML method or algorithm to reduce errors and generalize the representations learned from the data.
- *Model evaluation:* is evaluating the built model against certain criteria to assess its performance. Model performance is usually a function defined to provide a numerical value to help us decide the effectiveness of any model.
- *Hyperparameters tuning:* is choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value controls the learning process.
- *Model Deployment and monitoring:* is the method by which you integrate a machine learning model into an existing production environment to make practical business decisions based on data.
- *Model Monitoring:* is the close tracking of the performance of ML models in production.

B. CLOUD PROVISIONING SERVICES

Cloud computing is a widely adopted paradigm for the delivery of computing services. Leading cloud platforms such as AWS,[2] Google Cloud,[3] and Microsoft Azure[4] offer a variety of provisioning services that can be used for model serving. In addition, they provide several architectures with different access management. These are the list of the most common cloud computing architecture.

[2]https://aws.amazon.com/
[3]https://cloud.google.com/
[4]https://azure.microsoft.com/en-us/

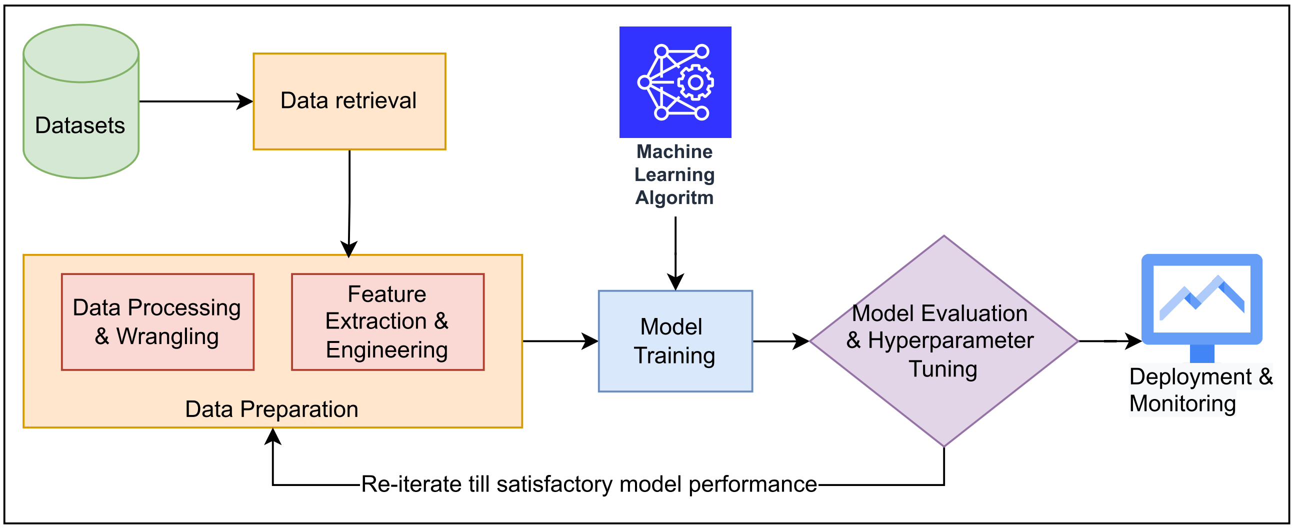
Datasets
Data retrieval
Machine Learning Algoritm
Data Processing & Wrangling
Feature Extraction & Engineering
Data Preparation
Model Training
Model Evaluation & Hyperparameter Tuning
Deployment & Monitoring
Re-iterate till satisfactory model performance

FIGURE 1. Typical ML pipeline [9].

- *Infrastructure-as-a-Service (IaaS).* It provides only the base infrastructure instances (VMs). The end-user must configure and manage the platform and environment and deploy applications.
- *Container-as-a-Service (CaaS).* It is a form of container-based virtualization in which container engines (e.g., Amazon ECS[5] and Google Kubernetes Engine[6]), orchestration, and the underlying computing resources are delivered to users as a service from the cloud provider.
- *Function-as-a-Service (FaaS).* With FaaS, customers run applications as serverless functions (e.g., AWS Lambda[7] and Google Cloud Functions[8]) and let the cloud platform to handle resource provisioning and management.

C. SERVERLESS COMPUTING
Serverless cloud computing is a model in which the service provider handles many tasks to ease certain burdens from the software developer(s). The provider is expected to automatically handle the necessary administration, deployment, and management tasks with scaling up/down resources. Furthermore, it is fully managed: engineers no longer have to worry about building and maintaining any underlying architecture and can delegate all responsibilities to the cloud vendor. Serverless users are billed by execution and resource consumption, not by an hourly or hardware-based rate [12]. However, compared to more traditional computing methods, serverless includes a laggy startup known as cold start [13].

III. STUDY DESIGN
In this research, we follow the guidelines for systematic mapping studies [14]. We present the procedure to review the literature on machine learning usage on serverless architecture. In the following, we present the design of our study, including the search keywords, search technique, data sources, and inclusion and exclusion criteria are explained.

A. RESEARCH QUESTIONS
We set the following list of research questions as a guideline during the systematic mapping review:

*RQ1 - WHAT ARE THE **PUBLICATION TRENDS** OF RESEARCH STUDIES ABOUT SERVERLESS ON MACHINE LEARNING?*

By answering this research question, we aim to characterize the intensity of scientific interest in using machine learning on top of serverless architecture, the relevant venues where academics publish their results on the topic and their types of contribution over the years.

*RQ2 - WHAT IS THE **FOCUS OF RESEARCH** OF APPLIED MACHINE LEARNING ON SERVERLESS COMPUTING ?*

By answering this research question, we aim to provide a solid foundation to classify existing research on machine learning in a serverless architecture.

*RQ3 - WHAT ARE THE **POTENTIAL CHALLENGES OF ADOPTING** MACHINE LEARNING ON SERVERLESS COMPUTING?*

By answering this research question, our objective is to profile the state-of-the-art on challenges and opportunities to use machine learning on serverless architecture.

B. DOMAIN EXPLORATION
In the following, we describe the interesting domain covered by this research during the systematic mapping study.

Cloud infrastructure: The types of cloud computing services vary, they provide access to IT infrastructure, hardware, and software resources. Cloud computing is all about delivering computing services like databases, software, analytics,

[5]https://aws.amazon.com/fr/ecs/
[6]https://cloud.google.com/kubernetes-engine
[7]https://github.com/aws/aws-lambda-go
[8]https://cloud.google.com/functions

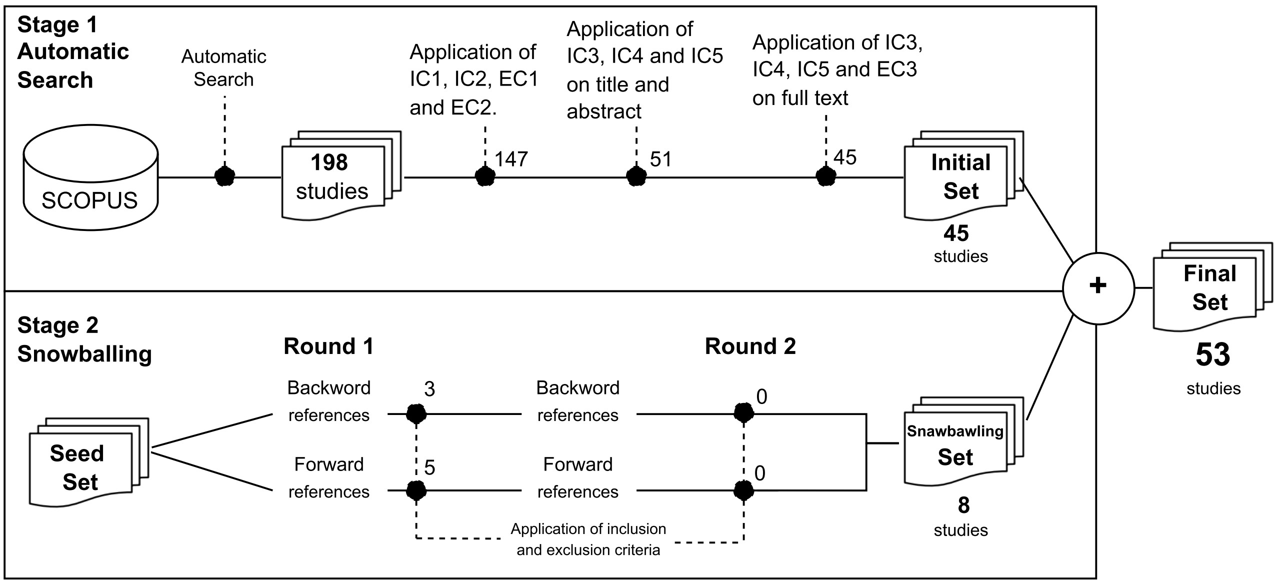

FIGURE 2. **Peer-reviewed selection process.**

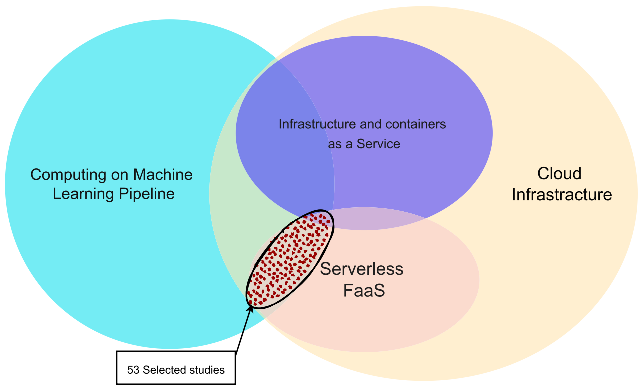

FIGURE 3. **Systematic literature mapping research focus.**

servers, storage, networking, and intelligence. There are many benefits of cloud computing, including cost savings, scalability, and access to data centers around the world [15]. Cloud computing services fall into four main categories: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS), and Functions as a Service (FaaS) which is a relatively new Cloud service model.

Computing on machine learning pipeline: cloud services are a good option for anyone looking to train and deploy memory-intensive, complex Machine Learning/Deep Learning models. Cloud services are a cost-effective solution for both individual users and companies. The cloud allows employees to access files on any device [16].

Serverless on Machine Learning: Serverless architecture gives many opportunities and advantages to make the machine learning model more efficient and smoother [17]. As shown in Figure 3, we focus on this systematic literature mapping on collecting research papers on applying serverless on machine learning.

C. SEARCH AND SELECTION PROCESS

As shown in Figure 2, we present our search and selection process. We designed a two-stage process, a systematic search similar to a previous study [18], to identify the current literature on serverless usage of machine learning. On Stage 1, we performed an automated search since it is the typical search strategy to identify relevant studies for a Systematic Mapping [19].

Defining the review goal, keywords were carefully selected to obtain relevant articles. In stage 1, several keywords were formulated and later narrowed down based on the research objectives. We designed our search query based on "machine learning" and "serverless." We executed the following search query on Scopus[9]:

```
(``serverless'' OR ``lambda
 architecture'' OR ``function as a
 service'') AND
(``machine learning'' OR ``deep
 learning'')
```

Since we are looking for a particular subject, we applied the default automatic search, including the title, abstract, and keywords. We executed the query in June 2022, where we found **198 studies**. The papers were either included among the relevant articles or excluded as irrelevant for the review by studying their titles, abstracts, conclusions and complete content.

To extract only relevant articles for review, certain inclusion (IC) and exclusion (EC) criteria were set, specifically:

- IC1: The study must be an article, conference paper, or workshop;
- IC2: The study must be in the Computer Science area;
- IC3: The study must be a primary study;

[9]https://www.scopus.com/

TABLE 1. List of included peer-viewed studies.

#	Reference	Title	Year
P01	[20]	A Case for Serverless Machine Learning	2018
P02	[21]	Exploring Serverless Computing for Neural Network Training	2018
P03	[12]	Implementation of unsupervised k-means clustering algorithm within amazon web services lambda	2018
P04	[22]	Pay-Per-Request Deployment of Neural Network Models Using Serverless Architectures	2018
P05	[23]	Serving deep learning models in a serverless platform	2018
P06	[24]	BARISTA: Efficient and scalable serverless serving system for deep learning prediction services	2019
P07	[25]	Behavior analysis using serverless machine learning	2019
P08	[26]	Cirrus: A Serverless Framework for End-To-end ML Workflows	2019
P09	[27]	Distributed Machine Learning with a Serverless Architecture	2019
P10	[28]	Function-as-a-Service Application Service Composition: Implications for a Natural Language Processing Application	2019
P11	[29]	Mark: Exploiting cloud services for cost-effective, slo-aware machine learning inference serving	2019
P12	[30]	On the FaaS track: Building stateful distributed applications with serverless architectures	2019
P13	[31]	Seneca: Fast and low cost hyperparameter search for machine learning models	2019
P14	[32]	Serving machine learning workloads in resource constrained environments: A serverless deployment example	2019
P15	[33]	Stratum: A serverless framework for the lifecycle management of machine learning-based data analytics tasks	2019
P16	[34]	Towards a Serverless Platform for Edge AI	2019
P17	[35]	TrIMS: Transparent and isolated model sharing for low latency deep learning inference in function-as-a-service	2019
P18	[36]	A cloud-based framework for machine learning workloads and applications	2020
P19	[37]	Automatic Tuning of Hyperparameters for Neural Networks in Serverless Cloud	2020
P20	[38]	Batch: Machine learning inference serving on serverless platforms with adaptive batching	2020
P21	[39]	Benchmarking Deep Neural Network Inference Performance on Serverless Environments With MLPerf	2020
P22	[40]	Enabling Cost-Effective, SLO-Aware Machine Learning Inference Serving on Public Cloud	2020
P23	[41]	FAASM: Lightweight isolation for efficient stateful serverless computing	2020
P24	[42]	Implications of Public Cloud Resource Heterogeneity for Inference Serving	2020
P25	[43]	Migrating Large Deep Learning Models to Serverless Architecture	2020
P26	[44]	Prognostics by classifying degradation stage on lambda architecture	2020
P27	[7]	Refactoring of Neural Network Models for Hyperparameter Optimization in Serverless Cloud	2020
P28	[45]	STOIC: Serverless Teleoperable Hybrid Cloud for Machine Learning Applications on Edge Device	2020
P29	[46]	Towards Federated Learning using FaaS Fabric	2020
P30	[6]	A Hybrid Framework for Effective Prediction of Online Streaming Data	2021
P31	[47]	A serverless gateway for event-driven machine learning inference in multiple clouds	2021
P32	[48]	AMPS-Inf: Automatic Model Partitioning for Serverless Inference with Cost Efficiency	2021
P33	[49]	Automatic Hyperparameter Optimization for Arbitrary Neural Networks in Serverless AWS Cloud	2021
P34	[50]	Cross-Platform Performance Evaluation of Stateful Serverless Workflows	2021
P35	[51]	Distributed double machine learning with a serverless architecture	2021
P36	[52]	Dorylus: Affordable, scalable, and accurate GNN training with distributed CPU servers and serverless threads	2021
P37	[53]	Edge-adaptable serverless acceleration for machine learning Internet of Things applications	2021
P38	[54]	Experience Paper: Towards enhancing cost efficiency in serverless machine learning training	2021
P39	[55]	FedLess: Secure and Scalable Federated Learning Using Serverless Computing	2021
P40	[56]	Gillis: Serving large neural networks in serverless functions with automatic model partitioning	2021
P41	[8]	High performance serverless architecture for deep learning workflows	2021
P42	[57]	Leveraging the serverless paradigm for realizing machine learning pipelines across the edge-cloud continuum	2021
P43	[58]	Performance and cost comparison of cloud services for deep learning workload	2021
P44	[59]	SLA-Aware Workload Scheduling Using Hybrid Cloud Services	2021
P45	[60]	Toward Sustainable Serverless Computing	2021
P46	[61]	Towards Demystifying Serverless Machine Learning Training	2021
P47	[62]	Towards situational awareness with multimodal streaming data fusion: Serverless computing approach	2021
P48	[63]	You Do Not Need a bigger boat: Recommendations at Reasonable Scale in a (Mostly) serverless and open stack	2021
P49	[64]	λDNN: Achieving Predictable Distributed DNN Training With Serverless Architectures	2021
P50	[65]	Serverless Computing Approach for Deploying Machine Learning Applications in Edge Layer	2022
P51	[66]	Stateful Serverless Computing with Crucial	2022
P52	[67]	INFless: A native serverless system for low-latency, high-Throughput inference	2022
P53	[68]	MLLess: Achieving Cost Efficiency in Serverless Machine Learning Training	2022

- IC4: The study should address machine learning practices using serverless;
- EC1: The study is not written in English;
- EC2: The study is duplicate;

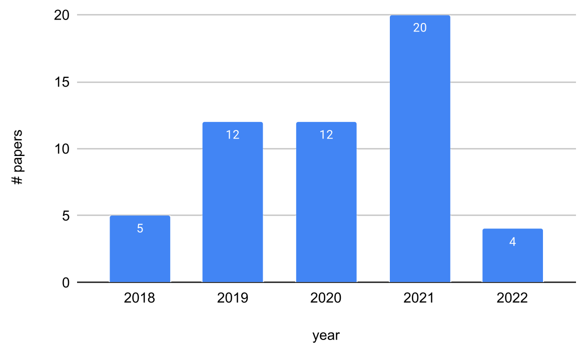

FIGURE 4. **Published papers per year.**

- EC3: The study is published as part of textbooks, abstracts, editorials, and keynote speeches.

After applying the IC1, IC2, EC1, EC2, we obtained **147 studies**. Then we analysed the title and abstract of each study and, after filtering by IC3, IC4 and EC3, we kept **51 studies**. We read all **51 papers**, filtering by IC3, IC4 and EC3 on the full text; we obtained the seed data set with **44 studies**. In Stage 2, we used our seed data set to perform two rounds of snowballing, backward and forward, detailed in 2. The snowballing research comes out with **8 additional studies**. Thus, we identify in the final dataset **52 studies** for this systematic mapping (Table 1). Our work is shredded on a public repository for study reproducibility.[10]

IV. RESULTS

The final set of publications presented in Table 1 was carefully read to answer the raised research questions. In the following, we are addressing carefully (1) the evolution trend of the set of papers and the different venues that were published; (2) the focus of the set of researchers on applying machine learning on Serverless architecture; (3) discussion of the challenges and opportunities to use Serverless on machine learning.

A. WHAT ARE THE PUBLICATION TRENDS OF RESEARCH STUDIES ABOUT SERVERLESS ON MACHINE LEARNING?

This research question aims at (1) characterizing the intensity of scientific interest and (2) the active publication venue on the usage of machine learning on serverless architecture.

1) PUBLICATION FREQUENCY

The selected papers of this study were analyzed to determine the trends in publication and the thematic evolution. Figure 4 shows the number of publications per year where researchers start exploring machine learning usage on serverless architecture. The results show that the average number of publications per year is approximately 12 from 2018 to 2021, starting with five papers in 2018 until 20 published papers in 2021.

[10]https://github.com/AmineBarrak/Serverless-on-ML

TABLE 2. **Number of published papers per each venue type.**

Venue type	#Studies	Studies
Conference papers	31	P02, P04, P05, P06, P07, P09, P11, P12, P13, P14, P15, P17, P19, P20, P23, P26, P27, P28, P32, P33, P35, P36, P38, P39, P40, P42, P43, P46, P48, P50 P52
Journal paper	10	P18, P21, P22, P30, P31, P37, P45, P49, P51, P53
Workshop paper	7	P01, P10, P16, P24, P29, P44, P47
Symposium paper	5	P03, P08, P25, P34, P41

TABLE 3. **List of journals (each of the following journals was mentioned once in the set of reviewed articles).**

Acronym	Journal Full Name
Concurr Comput	Concurrency and Computation: Practice and Experience
IEEE Access	IEEE Access
IEEE Software	IEEE Software
IEEE Trans. on Cloud Comput.	IEEE Transactions on Cloud Computing
IEEE Trans Comput	IEEE Transactions on Computers
J. Phys. Conf. Ser.	Journal of Physics: Conference Series
IEEE Internet Comput.	IEEE Internet Computing
Software Pract. Exper.	Journal of Software: Practice and Experience
ACM Trans. Softw. Eng. Methodol.	ACM Transactions on Software Engineering and Methodology
cs.DC	Distributed, Parallel, and Cluster Computing

Serverless computing has trended a significant engagement over the past years [2]. This boost has been caused by industry, academia, and developers for several reasons [69]. With the appearance of MLOps that include continuous and repetitive tasks (*i.e.,*code integration, training, deployment [70]), Serverless has started attracting ML developers.

2) PUBLICATION VENUE

Researchers have been contributing on the usage of serverless on ML pipelines. Table 2 shows the various publication venues we find in the selected research papers.

The percentages of publications in conference papers, journal papers, workshop papers, and symposium papers are approximately 60% (31/53), 16% (10/53), 14% (7/53), and 10% (5/53), respectively. The topic Serverless for ML practices has started attracting more researchers, we found ten journal papers that were published which reveal the subject relevance where more studies can present additional contributions.

Following the interpretation of publications, the most productive and primary journals, symposiums, conferences, and workshop venues related to serverless computing can be clarified. The list of journals we found is shown with their full names in Table 3. All eight journal venues were mentioned only once each.

The list of conferences is shown in Table 4. The "Cloud", "WOSC", "ICPE", "IC2E", "USENIX", "CCGRID",

TABLE 4. List of conferences.

Name	#Venue
International Conference on Cloud Computing (CLOUD)	3
International Workshop on Serverless Computing (WoSC)	3
International Conference on Performance Engineering(ICPE)	3
International Conference on Cloud Engineering (IC2E)	2
International Middleware Conference (Middleware)	2
USENIX Annual Technical Conference (ATC)	2
International Conference on Big Data (Big Data)	2
International Symposium on Cluster, Cloud and Internet Computing (CCGRID)	2
Service-Oriented Computing and Applications (SOCA)	1
International Conference on Prognostics and Health Management (ICPHM)	1
International Conference on Information and Communication Systems (ICICS)	1
Innovation in Clouds, Internet and Networks and Workshops (ICIN)	1
International Conference on Parallel Processing (ICPP)	1
USENIX Operational Machine Learning (OpML)	1
USENIX Hot Topics in Edge Computing (HotEdge)	1
International Symposium on Software Reliability Engineering Workshops (ISSREW)	1
Pervasive Computing and Communications (PerCom)	1
International Conference for High Performance Computing, Networking, Storage and Analysis (SC)	1
North American Chapter of the Association for Computational Linguistics (NAACL)	1
International Conference on Distributed Computing Systems (ICDCS)	1
High Performance Serverless Computing (HiPS)	1
International Conference on Computing for Sustainable Global Development (INDIACom)	1
International Conference on Management of Data (SIGMOD)	1
Symposium on Cloud Computing (SoCC)	1
International Conference on Software Engineering Workshops (ICSEW)	1
Big Data in Emergent Distributed Environments (BiDEDE)	1
Conference on Recommender Systems (RecSys)	1
Neural Information Processing Systems (NeurIPS)	1
International Symposium on Workload Characterization (IISWC)	1
International Conference on Information Networking (ICOIN)	1
Operating Systems Design and Implementation (OSDI)	1
Architectural Support for Programming Languages and Operating Systems (ASPLOS)	1

"Middleware", "ATC" and "Big Data" are considered the most active conferences held 19/42 (45%).

B. WHAT IS THE FOCUS OF RESEARCH OF APPLIED MACHINE LEARNING ON SERVERLESS COMPUTING ?

In this research question, our objective is to provide a solid classification of the existing research.

1) RESEARCH STRATEGIES OF SERVERLESS USAGE ON ML PIPELINE

Machine Learning pipelines are composed of four main stages, as shown in Figure 1 (1) data processing, (2) model training, (3) Hyperparameter tuning, and (4) model deployment. We examine the papers to determine the main goal and the main solution each study proposes. As shown in Table 5, the most recurrent usage targeted by the primary studies are in deploying ML models on Serverless (33/53), followed by model training (16/53), hyperparameter tuning (9/53) and data preprocessing (9/53). There are (6/44) studies that tried to employ Serverless in the end-to-end ML pipeline. These results confirm that the use of serverless benefits in the different stages of ML is advantageous.

TABLE 5. Serverless applied on machine learning pipeline.

ML pipeline	#Studies	Studies
Data preprocessing	9	P01, P07, P08, P10, P25, P26, P34, P46, P48
Training / Learning	16	P01, P02, P03, P08, P09, P23, P29, P30, P36, P38, P39, P46, P49, P50, P51, P53
Hyperparameter Tuning	9	P01, P02, P08, P13, P19, P27, P33, P35, P46
Model Deployment (inference)	33	P04, P05, P06, P10, P11, P12, P14, P17, P18, P20, P21, P22, P23, P24, P25, P28, P30, P31, P32, P34, P35, P37, P40, P41, P41, P42, P43, P44, P45, P47, P50, P51, P52
End-to-end ML pipeline	6	P01, P08, P15, P16, P34, P42

TABLE 6. Serverless platforms usage.

Serverless provider	**#Studies**	**Studies**
AWS Lambda	39	P01, P02, P03, P04, P05, P08, P09, P10, P11, P12, P13, P14, P19, P20, P21, P22, P24, P25, P26, P27, P31, P32, P33, P34, P35, P36, P37, P39, P40, P41, P43, P44, P45, P46, P47, P48, P49, P51, P52
Apache OpenWhisk	4	P18, P29, P39, P42
IBM Cloud Functions	4	P07, P38, P39, P53
Google Cloud Functions	3	P29, P39, P40
Azure Functions	3	P34, P39, P49
OpenFaaS	3	P29, P39, P52
Knative	2	P23, P50
KNIX	1	P40
Kubeless	1	P28

2) THE DIFFERENT SERVERLESS PROVIDERS

Table 6 presents the serverless platforms used in the considered research papers included in this study. It can be noticed that "AWS Lambda" has significant usage in 39 studies. We also found that "Apache OpenWhisk", "IBM Cloud Function" and "Google Cloud Function" are used with 4, 4, and three published papers, respectively. Each platform has its own set of features and differs from others. We later compare the different providers in RQ3 IV-C.

3) THE MAIN RESOLVED/DISCUSSED CHALLENGES AND ISSUES

The main solved / discussed challenges are cost / pricing (37/53) and resource scalability (30/53), as reported in Table 7. The high number of studies that discussed (1) cost/price and (2) scalability might indicate that Serverless provides a fair price architecture that provides a pay-per-use model that auto-scales in needs. Researchers seem to be interested in using Serverless for model deployment and make sure to keep a rational inference latency (22/53),

TABLE 7. Machine learning & serverless challenges and issues.

ML & serverless Challenges and Issues	#Studies	Studies
Cost and pricing	37	P02, P03, P04, P05, P06, P08, P09, P10, P11, P12, P14, P15, P17, P20, P22, P23, P24, P25, P30, P31, P32, P33, P34, P35, P36, P38, P39, P40, P41, P43, P44, P46, P47, P49, P51, P52, P53
Resources scalability	30	P01, P04, P05, P08, P11, P12, P14, P17, P18, P19, P22, P23, P25, P27, P29, P31, P33, P34, P35, P36, P38, P39, P41, P43, P44, P46, P47, P51, P52, P53
Inference latency	22	P01, P04, P05, P06, P11, P15, P16, P17, P20, P21, P22, P23, P24, P25, P28, P30, P34, P40, P42, P47, P50, P52
Batching (varying batch size)	16	P03, P08, P09, P11, P17, P20, P25, P26, P28, P32, P37, P38, P41, P49, P52, P53
Cold Start	10	P05, P17, P21, P23, P25, P28, P34, P40, P41, P52
Service Level Objective (SLO)	10	P06, P11, P20, P22, P24, P32, P40, P44, P45, P52
Edge Computing	6	P15, P16, P28, P42, P45, P50
Storage Latency	6	P08, P12, P14, P26, P41, P51
Security	4	P23, P29, P33, P39
Privacy	4	P16, P29, P39, P50
Training Latency	2	P02, P34
End-to-end Latency	2	P37, P50
Network Latency	2	P17, P36
Portability	2	P18, P31

TABLE 8. Machine learning frameworks used in the studies.

ML frameworks	#Studies	Studies
Tensorflow	22	P02, P08, P09, P11, P17, P19, P20, P21, P22, P23, P24, P25, P27, P28, P29, P32, P37, P39, P41, P49, P50, P52
Keras	10	P11, P19, P22, P27, P28, P32, P33, P37, P39, P42
MXNet	9	P05, P09, P11, P16, P17, P20, P22, P24, P40
Pytorch	8	P04, P14, P36, P38, P43, P44, P46, P53
Scikit-learn	6	P10, P28, P34, P35, P37, P42
Numpy	5	P10, P23, P35, P42, P43
PyWren	4	P01, P08, P12, P38
Spark ML	3	P08, P12, P26
OpenCV	3	P21, P25, P41
ONNX	2	P04, P14
Tesseract	2	P25, P41
Pandas	2	P35, P42
Bosen	2	P01, P08
Pillow [73]	1	P32
Caffe	1	P17
MNIST	1	P50

in contracts (6/53), (2/53) and (2/53) focused on the storage, network, and training latencies, respectively. There were proposed solutions to reduce latency and improve performance by varying the batch size; this solution was present in (16/53). The cold start was discussed in (10/53) studies trying to

TABLE 9. Type of machine learning algorithms used to train models.

Types of ML models	#Studies	Studies
Neural Network (DNN, CNN, RNN, GNN)	22	P02, P04, P13, P14, P17, P18, P19, P21, P27, P28, P29, P32, P33, P36, P37, P40, P42, P43, P44, P45, P49, P50
Supervised ML (LR, RF, SVM)	14	P01, P06, P07, P08, P12, P26, P30, P31, P34, P35, P42, P46, P51, P53
ResNet	12	P05, P06, P11, P17, P20, P22, P24, P32, P40, P49, P46, P52
Inception	8	P06, P11, P17, P20, P22, P24, P32, P40
LSTM	7	P11, P14, P19, P22, P30, P39, P52
Stochastic Gradient Descent (SGD)	6	P01, P09, P23, P38, P46, P53
MobileNet	5	P20, P32, P49, P46, P52
squeezenet	3	P05, P17, P24
NLP	3	P04, P10, P21
K-means	4	P03, P12, P46, P51
VGG	3	P06, P33, P52
Reinforcement Learning (RL)	2	P09, P40
Federated Learning	2	P29, P39
NASNet	2	P11, P22
OpenNMT	2	P11, P22
Optical Character Recognition (OCR)	2	P25, P41
Yolo	2	P31, P47
Gym openAI	1	P09
Bert	1	P52

mitigate it since Serverless containers have start-up latencies in the hundreds of milliseconds to several seconds, leading to the cold-start problem [71]. A significant number of studies (10/53) discussed the Service Level Objective (SLO). We mention that the SLO is an agreement set by a Serverless provider where there is the pre-defined service minimum response time [40]. Interestingly, few papers considered security and privacy with (4/53) and (4/53), respectively. However, only (2/53) paper mentioned the portability and reproducibility of the run-time environment.

4) MACHINE LEARNING FRAMEWORKS USED IN THE STUDIES

The machine learning frameworks helped the researchers to test their proposed solution easily, without understanding the underlying algorithms. Therefore, the choice of framework depends on the complexity of the targeted task. As reported in Table 8, the predominant ML frameworks are: Tensorflow (22/53), Keras (10/53) and MXNet (9/53). Indeed, other frameworks have been used in recent studies such as Pytorch (8/53) since it can be used for distributed training in parallel machines [61], Numpy (5/53) and OpenCV (3/53).

5) TYPE OF MACHINE LEARNING ALGORITHM USED TO TRAIN MODELS

The type of machine learning used to train the models depends on the research goals. Table 9 shows what type of

TABLE 10. Comparison between the most known providers for Function as a Service (Serverless).

Serverless Provider	Function timeout (second)	Maximum Memory Allocation	Deployment package size
AWS Lambda [75]	900	10,240 MB	50MB zip 250 MB unzip
Apache OpenWhisk	600	2048 MB	48 MB
Google Cloud Functions	540	8560 MB	100MB zip 500MB unzip
IBM Cloud Functions	600	2048 MB	48 MB
OpenFaaS	300	42 MB	1MB
Azure Functions	600	1500 MB	n/a
KNIX [76]	30	2000 MB	100 MB
Knative	600	200 MB	256 MB unzip

machine learning was used. The results show that neural network models dominate the studies with (22/53). Neural network models are more challenging to be used, especially in distributed environment *i.e.,*distributed ML training [54]. Surprisingly, we found the use of supervised ML, such as logistic regression, random forest, and SVM, in (14/53) studies. These models are not resources costly in the training phase. Their usage is mostly for comparison purposes of the proposed architecture [30]. There are several other machine learning algorithms. We mention ResNet (12/53) and Inception with different versions (8/53). These models are based on a conventional neural network used for intensive computing *i.e.,*image recognition [73].

C. WHAT ARE THE POTENTIAL CHALLENGES OF ADOPTING MACHINE LEARNING ON SERVERLESS COMPUTING?

In this research question, our objective is to profile state of the art on challenges of machine learning usage on serverless architecture.

1) SERVERLESS PROVIDERS

It is interesting to see Serverless providers evolving their services over the years. Carreira *et al.* [20] discussed about the Serverless capacity as they were not able to run Tensorflow [76] or Spark [77] functions on AWS lambda due to size limits (3GB RAM). Today the limit RAM size has increased to 10 GB for each serverless function [78]. We present in Table 10 the Serverless performance functionalities offered by the different providers we found in the primary studies. This table was filled in January 2022. We did not include Kubeless in the comparison table, since it is not an active project. In the previous RQ, we found that 77% of the studies (34/44) were using AWS Lambda. We can explain that result because this tool provides a high Random Access Memory allocation to reach 10Gb. Moreover, since the serverless function works only on demand, it has a timeout where the instance is shutdown after a timeout set by the provider. We can see that Amazon has the longest function timeout.

We noticed that the deployment package size is small and differs from one provider to another. It is the total size allowed for the function source code *i.e.,*model. Providers offer the possibility of hosting the deployed model in extra storage if the model exceeds the limit for the serverless package size. For example, there is an option to use an external database or S3 bucket to store large payloads and pass the data identifier to the function calls. However, this option will cause additional latency to the system.

For better services, the serverless providers may ensure better user performance, especially the timeout function, to keep the instance warm and avoid the cold start latency.

2) SERVICE LEVEL AGREEMENT/OBJECTIVE

A Service Level Agreement/Objective is an agreement set by the serverless provider. Cloud providers claim different SLAs due to their unique technics. In such case, the performance may vary for the same code from one cloud service provider to another [40]. We found that all the studies discussing the SLO agreement use the serverless for ML model deployment [24], [29], [38], [40], [42], [48], [56], [67]. They ensured that the inference model in their proposed solution was respected. For example, Amazon SLO regarding inference latency is that at least 98% of inference queries must be served in 200 ms. However, failing to acquiesce with the SLOs results will lead to compromised quality of service or even financial loss, e.g., end users will not be charged for queries not responded in time [79]. Regarding the machine learning models inference, the execution of small models (e.g., MNIST, Textcnn-69) can respond within 50ms under each memory configuration, but for the other large models, such as Bert-v1, ResNet-50 and VGGNet, a small memory configuration leads to quite a long execution time (exceeding hundreds of milliseconds). If configured with the maximum allowable memory size, the execution time for a single request exceeds 200ms, which makes it challenging to meet the latency SLO in the production environment [67]. Therefore, providers should share such agreements and statistics of service violations to help customers choose the best one, leading to a competitive environment for better services.

3) ENSURE RESOURCE SCALABILITY AND PREDICTIVE SCALING

In general, serverless architecture provides autoscaling features to handle workload spikes smoothly. Forecasting resource usage is no longer necessary to ensure that we always have the right amount of resources to host our applications. Compared with cluster computing, a serverless base model enables a rapid adjustment on-demand of the number of workers overtime [61]. Moreover, in multithread computation, a single-machine solution quickly degrades when the number of threads exceeds the number of available cores, while in a serverless base solution, the scale-up is faster regarding the execution time [66].

However, serverless functions do not support customized scaling. Barista uses predictive scaling to achieve low-latency inference serving in the serverless cloud [24]. Moreover, a hybrid architecture between Serverless and VM is proposed, and even machine learning models to predict scaling to reduce over-provisioning to the best execution environment [29], [40].

The machine learning inference services are commonly latency critical, and the auto-scaling ability of serverless computing could deal with bursty workloads well [8]. Yang *et al.* [67] presented a solution called INFLess that reduces the allocation of resources for each serverless instance to reach optimal performance in inference services.

The resource scalability is a critical property of any ML training system since only the active workers at any given time will be billed. ML training is typically an iterative process in which a higher number of workers is desirable during the first training steps to diminish loss. When loss reduction stagnates and reaches convergence, the number of workers scales down once the learning curve starts to flatten out [54], [68].

4) SERVERLESS VS. IaaS FOR ML SERVING

Several works in primary studies (37/53) developed the idea of reducing costs by using serverless in their ML solution (deployment, testing, etc.) instead of an infrastructure environment.

Serverless providers are proposing pay-as-you-go services, only paying for those resource usage - compared to other cloud resources such as the IaaS compute service AWS EC2, where customers would be paying for the instance even when there is no traffic. Concretely, it is not all the time that Serverless is better than IaaS. When a company's traffic is known to deploy their model online, they must choose the service that performs their business model. If the traffic is unknown, it is better to use serverless since the payment is only for executions [80]. A hybrid architecture can be considered as an additional solution [29].

5) COLD START

Cold start in serverless is somehow expensive and causes significant performance degradation for serverless functions [81]. However, it is still better than other cloud resources *i.e.,*VM. There were several solutions for the cold start, for example, periodically warming the instance [56] or predicting the window timing where a request is expected, or scheduling the tasks [45]. To reuse against the cold start, in [28] presented a switchboard architecture composed of 6 serverless with the principle to warm the first function and trigger the rest of the functions. Another proposed to keep the instance warm for several minutes [82]. In [29], authors showed that keeping the instance warm has a low cost. They explained how $1 could spin up 7K inception-v3 Lambda instances, which can serve more than 20K requests per second. To avoid the cold start latency, Yang *et al.* [67] proposed a Long-Short Term Histogram (LSTH) to track application idle times and draw two histograms. The two histograms represent the request patterns in the last short (e.g., 1 hour) and long durations (e.g., one day). By tracking the application, they can select the pre-warming window to send inference requests to continuously keep the function instance alive. Their method helped to reduce resource waste while avoiding cold starts.

6) SECURITY AND PRIVACY

Privacy and security are always major concerns in serverless computing, especially for managing and analyzing sensitive data, such as healthcare data.

We observed that it is essential to set roles for every cloud function with specific security policies to provide only necessary access and prevent non-permitted operations. For example, Kaplunovich and Yesha [49] applied special protection to the hyperparameter metadata spreadsheet, where metadata is loaded directly during the startup and stored safely and securely in the protected Cloud location.

The Federated Learning-based architecture was proposed in the primary dataset [46], [55], [65]. This computing model supports edge computing, where the processing edges can learn from a shared machine learning model while keeping the model training on remote clients, followed by global aggregation of the updated model parameters. This keeps the training data local, which provides privacy and security benefits. Grafberger *et al.* [55] considers that the challenges of FL systems, such as scalability, complex infrastructure management, and wasted computing, can be solved with the Function-as-a-Service (FaaS) paradigm. However, it is necessary to be aware of the threats caused by malicious participants. For example, Tolpegin *et al.* [83] showed that a malicious subset of participants could decrease the accuracy of the model by injecting poisoned data when sending updates to the global model.

Several additional security measures can be applied, where only authorized and authenticated entities can invoke client functions. A practice of security between clients was applied in [55], where the FL server allows clients authenticated to read only from a shared global model and write back results without access to other clients. Another security measure was applied in [55], where HTTP function requests exchanges can be encrypted using Transport Layer Security (TLS).

Rausch *et al.* [34] chose to transmit the base model to an edge device to refine the base model locally using a serverless function with the private data to ensure data privacy. The edge computing paradigm allows training distributed machine learning models between local edge data to secure data privacy and save resources in the cloud [65]. Bac *et al.* [65] applied a federated learning approach on serverless edge computing, where they saved bandwidth and ensured the data privacy of the edge nodes.

Moreover, Anthony S. Deese [12] handled the used access by applying AWS Cognito and Identity Access Management services. These services allow a user to access and monitor only the lambda function instances he created, which maintains the privacy of user training data and machine results.

7) BATCHING

Another essential factor that heavily impacts both the cost and the performance of ML serving inference is batching. For example, the batch size cannot be arbitrarily increased, as it leads to longer queuing latency and batch inference latency [84]. Tuning batch or resource configuration adaptively can improve the model performance. Clipper [84] introduces caching, batching, and adaptive model selection techniques to reduce the latency. INFaaS [85] automatically adapts the model variant batch size and hardware according to the required model performance.

For smaller batch sizes, the processing time increases linearly, but for larger batch sizes, it increases exponentially in an on-premise environment [8]. Deese [12] used the batch mode (maximum size) read and write requests within AWS and found a significant speed increase from batch write operations, but a relatively small benefit from batch reads. Carreira *et al.* [26] finds that data fetching latency becomes low when applied mini-batches buffers. Wang *et al.* [27] considered that machine learning serverless functions should have a different size of data batch since many training samples need to be processed by different workers in parallel. Zhang *et al.* [29] showed that inference serving could benefit significantly from batching using costly hardware accelerators (e.g., GPU and TPU). The appropriate batch size with GPU instances can achieve a lower cost and shorter inference latency. However, serving inference queries using GPUs is not economically justified when there is not enough load. MLLess [68] kept the same mini-batch size in their distributed workers architecture to avoid changing the number of workers incurring costly data repartitioning transfers to adjust the mini-batch size.

Depending on the usage purpose of serverless computing in the machine learning pipeline phase, batching size can play an important role in reducing processing time, deployment latency, and data fetching.

8) SERVERLESS PORTABILITY

When using platform services from public cloud providers, there is a risk of dependence on the services and products they offer. This case is named the 'vendor lock-in' since switching technologies and vendors can be costly due to technical incompatibilities. Naranjo *et al.* [47] proposed to use open-source frameworks instead of public cloud providers. Most open-source serverless platforms rely on Kubernetes for orchestration and management of function pods, which makes the portability task more affordable [86]. Unfortunately, portability did not take enough chances in the set of the studied papers. We found only two journal papers [36], [47] that took into account the level of portability of the run-time environment.

Portability is an important aspect; only when portability is ensured is a helpful simulation to test if the function shows better performance on another platform [69]. The Serverless Framework [87] offers plug-ins to simplify the deployment and execution of serverless functions across multiple clouds and FaaS environments. Junfeng Li *et al.* [86] compared the performance of four open-source serverless platforms using CloudLab testbed. Their work was provided to help developers to differentiate and select the appropriate serverless platform for different demands and scenarios.

9) EDGE COMPUTING

Edge computing is a distributed computing paradigm that brings computation and data storage closer to the data sources, especially popular with IoT device architecture. Edge computing has several benefits, such as reducing latency and bandwidth associated with public cloud [53], ensuring data privacy [34], and reducing computational resources relative to public and private clouds [45].

The serverless edge computing platform that provides the appropriate support to define AI workflow functions has been extended to work at the edge of the network to reduce response latency and bandwidth associated with the public cloud [34], [53], [65].

The ML module can be placed on the edge devices, or it can be placed on the Cloud or Fog layer for live or in-depth analysis of the data [33]. Zhang *et al.* [45] proposed a hybrid cloud system consisting of edge and cloud resources and integrating GPU acceleration. The usage of edge computing depends on the user requirements and the analysis of available capacity.

10) COST REDUCTION

One of the primary purposes of using serverless with machine learning is cost reduction. High service costs are the major issue that papers try to reduce in different ways. Serverless usage is adopted to reduce unnecessary costs and improve manageability, like the allocation of virtual machines without full resource usage. For example, Wang *et al.* [27] demonstrated that a substantial amount of cost savings can be achieved by replacing dedicated IaaS cloud clusters with a serverless architecture. They proposed a solution called SIREN to reduce the training cost compared MXNet architecture. The AMPS-Inf achieves up to 98% cost savings without degrading response time performance [48]. Chahal *et al.* [59] presented an architecture based on load balancing the ML inference workload to reduce costs.

Cost reduction is a primary concern for developers and researchers. The cost is related to the design architecture, computing, inference deployment, and read/write queries. Depending on the machine learning project, a serverless-based architecture could be an effective option to reduce the cost.

11) INFERENCE LATENCY

Inference latency was well studied in the primary set of papers. Yu *et al.* [56] showed that the inference latency increases as the model grows. They proposed a serving model and generated a parallelization scheme deployed on serverless platforms to achieve optimal inference latency.

Zhang *et al.* [39] required a benchmark analysis to find an efficient inference workload. They found that the amount of memory allocated for each serverless function instance plays an important role in inference latency time reduction.

The latency can be influenced by the serverless cold start, or continued serverless warming [43]. Moreover, the hardware usage, such as GPU instances with the appropriate batch size, can have shorter inference latency compared to the CPU instances [29]. Gujarati *et al.* [88] proposed an autoscaling framework that aims to minimize resource waste for ML inference by using a predictive provision model. BATCH [38] designed a buffer layer on top of the serverless platform and bundles requests with batching for cost-saving serverless inference. Moreover, inference latency can be dominated by data fetching when there are queries involving cross-machine requests [22].

12) MLOps AND SERVERLESS

The MLOps is modeled to make the intersection between machine learning, data engineering, and DevOps practices that associate software developers (the Devs) with IT operations teams (the Ops) to collaborate [89].

The machine learning pipeline contains several repetitive steps (data collection, data integration, data preparation and cleaning, model retraining, predictions) that need special operations to be automated in MLOps environments. The serverless architecture can be used to (1) automate the infrastructure; (2) build event-driven applications; (3) build APIs *i.e.,* API with Amazon gateway.

Serverless with data preprocessing: The serverless can be scheduled to pull data from the backend; trigger a serverless function when objects are written in data buckets *i.e.,* AWS S3; build APIs to transform and clean data.

Serverless with model retraining: Schedule or trigger new training when conditions are met.

Serverless with model inference: Schedule a serverless function for batch predictions; use step functions for ensemble predictions.

The serverless architecture can be feasible and optimal for projects adapting the MLOps approach. We plan as future work to explore how serverless can fit and optimise the MLOps-based projects.

V. RELATED WORK

This section presents related work that realised a literature review on serverless computing and cloud computing applied to machine learning.

A. LITERATURE REVIEWS ON SERVERLESS

serverless computing is getting more popular. Wen *et al.* [2] mined and analyzed serverless-based questions from Stack Overflow to show an increasing popularity trend of the subject and presented a list of challenges that present an overhead for developers during the usage of serverless computing, such as programming language support, database connection, and Resource Configuration to Security.

Several studies have searched on serverless challenges. For example, Khatri *et al.* [90] presented a review of the potential bottleneck and measured the performance of serverless computing. Their work was more related to serverless limitations such as peak and spike scenarios, scalability, cold start, and portability. They showed especially the difficulties of testing and performance measurement of serverless applications and how machine learning can monitor and predict performance. Moreover, Hassan *et al.* [69] applied a survey including 275 research papers that examined the challenges that serverless computing faces nowadays and how future research could enable its implementation and usage. Furthermore, Wu *et al.* [91] presents several practical recommendations for data scientists on using serverless for scalable and cost-effective model serving.

The main challenge in serverless computing is reproducibility. Scheuner and Leitner [92] conducted a multivocal literature review on the evaluation of function as a service performance, covering 112 studies. They evaluated these studies from the reproducibility perspective and found that most studies do not follow reproducibility principles in cloud experimentation. More challenges were discussed by Sadaqat *et al.* [93]. They conducted a multivocal literature review to define the core components of serverless computing, its benefits, and its challenges. They found that serverless computing is a solution that allows users to create functions that intercept and operate on data flows in a scalable manner without the need to manage a server, discussing that vendor lock-in, skilled workers, testing complexity, and monitoring are the most recurrent challenges. They also presented the expected evolution of serverless computing, such as the adoption of serverless by companies and the expected market growth.

Tiabi *et al.* [94] identified 32 patterns composing and managing serverless functions by applying for a multivocal literature review on 24 selected papers. They classified the patterns as orchestration, aggregation, event management, availability, communication, and authorization. They show that depending on the serverless provider. The pattern may not be the same, *i.e.,* AWS lambda adapted their queue service (SQS) to enable FIFO messages. However, FIFO messages still need to be manually managed in Azure. They present their work as a pattern catalog that provides a valuable basis for practitioners and researchers on serverless computing.

The different challenges identified in the literature related to the serverless were discussed in our set of papers on machine learning perspectives.

B. LITERATURE REVIEWS ON MACHINE LEARNING & CLOUD COMPUTING

Machine learning has been used in several architectures of cloud computing. John *et al.* [95] conducted a systematic literature review of 13 primary studies related to AI deployment in the context of edge/cloud/hybrid architectures. They presented a list of practices and challenges of practitioners related to the design, integration, deployment, operation, and model evolution. They conclude their work by proposing an end-to-end model deployment framework. Moreover, Kuhlenkamp *et al.* [96] conducted a systematic literature review on machine learning operationalization. They investigate the techniques, tools, and infrastructures to operationalize ML models. They reviewed 24 studies that discussed the presence of several tools for model operationalization *i.e.,*Polyaxon,[11] and MLflow.[12] They found that cloud computing is widely used in model deployment due to resource hardware heterogeneity and the possible variety of network connection quality. Furthermore, Jauro *et al.* [97] realised a survey on the usage of deep learning algorithms to solve complex problems in emerging cloud computing architectures. Their study included 34 studies focusing on edge, fog, serverless, volunteer, and software-defined computing. During their study, they identified the strengths and limitations of the different deep learning algorithms regarding their suitability to the problem of solving *i.e.,*image processing, time series, and regression. Distributed machine learning is widely used, especially with IoT devices. Filho *et al.* [98] realised a systematic literature review on 106 research papers about the distributed machine/deep learning intelligent algorithms in edge devices *i.e.,*IoT. They investigated the challenges of running ML/DL on edge devices in a distributed way, such as limited resources, communication efficiency, and ensuring data privacy and security. They found several techniques to mitigate the challenges related to edge computing *i.e.,*caching, training, inference, and offloading. Setti Cassel *et al.* [99] analyzed 60 research papers on serverless IoT devices during a systematic literature review. They find that serverless computing is a promising technology for IoT applications that can bring functions closer to the devices to reduce latency and avoid unnecessary energy and resource consumption.

In this work, we focus mainly on studies realised on machine learning usage on top of serverless computing architecture.

VI. THREATS TO VALIDITY

We applied Peterson guidelines to make our systematic mapping study [14]. However, threats to validity are unavoidable. This section presents the main threats to the validity of our study and how we mitigated them.

External validity.. External validity relates to the generalizability of our results. The most severe external threat is that finding all the relevant studies on machine learning applied to serverless architecture from the designed query is absurd. As a solution, we applied a search strategy to the initial set of papers consisting of both automatic search and recursively backward-forward snowballing. Additionally, we applied a well-established peer-reviewed analysis to ensure that we have high-quality publications. We carefully defined the inclusion/exclusion rules that respect the requirements of our study with the agreement of all authors.

Internal validity. Internal validity relates to the experiment errors and biases. We mitigate the internal validity threats caused by author bias when selecting and interpreting data by applying well-assessed descriptive statistics of the collected data. Several re-verification steps between authors were performed to ensure a good classification dataset.

Construct validity. Construct validity is related to the degree to which an evaluation measures what it claims. We mitigated this potential bias by carefully defining the research query on the Scopus database. This database was preferred since it offers a more extensive list of modern sources [100]. In the keywording process, we included different taxonomies that can be mentioned to refer to the serverless, *i.e.,*lambda architecture, function as a service. Also, we are fairly confident about constructing the search string since the automatic search has been followed by snowballing. Also, we rigorously selected the potentially relevant studies according to well-documented inclusion and exclusion criteria. The first author performed this selection stage, and randomly a sample set was verified by the second author and agreement was ensured.

Conclusion validity. Conclusion validity is related to random variations and inappropriate use of statistics. To mitigate it, we rigorously defined and iteratively refined our classification framework, such as suggested by [101], so that we could reduce potential biases during the data extraction process. In addition, we ensured that we aligned with our research question and our main research objectives. We mitigated potential threats to conclusion validity by applying the verification agreement between authors in case of disambiguating cases. We provide a public repository for the reproducibility of the study to determine whether other researchers could obtain similar results from this study.[13]

VII. CONCLUSION

This study aims to provide a broader survey investigating the relationships among research contributions on Machine Learning usage on Serverless architecture. Specifically, we performed a systematic mapping on 50 primary studies and produced an overview of the state of the art on machine learning applications on serverless architecture. We found that (1) serverless usage on machine learning applications is a growing field starting from 5 on 2018 until 20 published papers on 2021, and more publication venues are interested to the subject; (2) serverless was adopted on the different ML pipeline, especially on ML model deployment with 33/53 papers. The most used serverless provider is usually AWS lambda, and the used ML model was the neural network. The main challenge of using serverless on ML was reducing cost and pricing (37/53), ensuring enough scalable resources (30/53), and reducing inference latency (22/53). There are several potential challenges of adopting ML on serverless,

[11]https://polyaxon.com/

[12]https://mlflow.org/

[13]https://github.com/AmineBarrak/Serverless-on-ML

such as respecting the service level agreement, serverless provider, cold start problem, security and privacy, serverless portability, resource scalability, batch size, edge computing, cost reduction, and inference latency.

Depending on the targeted architecture and the solution, a trade-off between inference latency, serverless cold start, cost, scalability, batch size, and portability must be considered. For example, an open source provider would be a good solution if portability is essential. The results of this study will benefit both researchers willing to contribute further to the area and practitioners willing to understand existing research.

In future work, we plan to explore the effectiveness of serverless benefits with MLOps practices, especially in a distributed computing environment. Moreover, hybrid cloud architecture for machine learning pipeline phases can be a subject to study its validity depending on the user objectives and their data flow type.

REFERENCES

[1] (Jul. 2018). *13 Benefits of Cloud Computing for Your Business | Globaldots*. Accessed: Apr. 16, 2022. [Online]. Available: https://www.globaldots.com/resources/blog/cloud-computing-benefits-7-key-advantages-for-your-business/

[2] J. Wen, Z. Chen, Y. Liu, Y. Lou, Y. Ma, G. Huang, X. Jin, and X. Liu, "An empirical study on challenges of application development in serverless computing," in *Proc. 29th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng.*, Aug. 2021, pp. 416–428.

[3] ReportsandData. (Oct. 2021). *Function-as-a-Service (FAAS) Market Size Worth USD 31.53 Billion at CAGR of 32.3%, by 2026—Report and Data—EIN Presswire*. Accessed: Jan. 20,2022. [Online]. Available: https://www.einnews.com/pr_news/552783688/function-as-a-service-faas-market-size-worth-usd-31-53-billion-at-cagr-of-32-3-by-2026-report-and-data

[4] M. Ribeiro, K. Grolinger, and M. A. M. Capretz, "MLaaS: Machine learning as a service," in *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2015, pp. 896–902.

[5] P. Castro, V. Ishakian, V. Muthusamy, and A. Slominski, "The rise of serverless computing," *Commun. ACM*, vol. 62, no. 12, pp. 44–54, Nov. 2019, doi: 10.1145/3368454.

[6] K. Kanagaraj and S. Geetha, "A hybrid framework for effective prediction of online streaming data," *J. Phys., Conf.*, vol. 1767, no. 1, 2021, Art. no. 012016.

[7] A. Kaplunovich and Y. Yesha, "Refactoring of neural network models for hyperparameter optimization in serverless cloud," in *Proc. IEEE/ACM 42nd Int. Conf. Softw. Eng. Workshops*, Jun. 2020, pp. 311–314.

[8] D. Chahal, M. Ramesh, R. Ojha, and R. Singhal, "High performance serverless architecture for deep learning workflows," in *Proc. IEEE/ACM 21st Int. Symp. Cluster, Cloud Internet Comput. (CCGrid)*, May 2021, pp. 790–796.

[9] *Standard ML Workflow | Hands-on Transfer Learning With Python*. Accessed: Jan. 26, 2022. [Online]. Available: https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781788831307/1/ch01lvl1sec05/standard-ml-workflow

[10] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," in *Proc. IEEE/ACM 41st Int. Conf. Softw. Eng., Softw. Eng. Pract. (ICSE-SEIP)*, May 2019, pp. 291–300.

[11] *MLOPS : Pipelines De Livraison Continue et d'Automatisation Dans Le Machine Learning | Google Cloud*. Accessed: Mar. 9, 2022. [Online]. Available: https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

[12] A. Deese, "Implementation of unsupervised K-means clustering algorithm within Amazon web services Lambda," in *Proc. 18th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGRID)*, May 2018, pp. 626–632.

[13] P. Silva, D. Fireman, and T. E. Pereira, "Prebaking functions to warm the serverless cold start," in *Proc. 21st Int. Middleware Conf.*, Dec. 2020, pp. 1–13.

[14] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Inf. Softw. Technol.*, vol. 64, pp. 1–18, Aug. 2015.

[15] *The 4 Types of Cloud Computing Services | Exitcertified*. Accessed: Jul. 13, 2022. [Online]. Available: https://www.exitcertified.com/blog/4-cloud-computing-services

[16] *Introduction to Cloud Computing for Machine Learning Beginners*. Accessed: Jul. 13, 2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2022/01/introduction-to-cloud-computing-for-machine-learning-beginners

[17] *How Serverless Architecture Can Impact the Future of AI and ML Industries | Engineering Education (Enged) Program | Section*. Accessed: Jul. 13, 2022. [Online]. Available: https://www.section.io/engineering-education/how-serverless-architecture-can-impact-the-future-of-ai-and-ml-industries/

[18] E. Mendes and F. Petrillo, "Log severity levels matter: A multivocal mapping," 2021, *arXiv:2109.01192*.

[19] S. Keele, "Guidelines for performing systematic literature reviews in software engineering," EBSE, Tech. Rep., Version 2.3, 2007.

[20] J. Carreira, P. Fonseca, A. Tumanov, A. Zhang, and R. Katz, "A case for serverless machine learning," in *Proc. Workshop Syst. ML Open Source Softw. (NIPS)*, 2018, pp. 1–7.

[21] L. Feng, P. Kudva, D. Da Silva, and J. Hu, "Exploring serverless computing for neural network training," in *Proc. IEEE 11th Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2018, pp. 334–341.

[22] Z. Tu, M. Li, and J. Lin, "Pay-per-request deployment of neural network models using serverless architectures," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Demonstrations*, 2018, pp. 6–10.

[23] V. Ishakian, V. Muthusamy, and A. Slominski, "Serving deep learning models in a serverless platform," in *Proc. IEEE Int. Conf. Cloud Eng. (ICE)*, Apr. 2018, pp. 257–262.

[24] A. Bhattacharjee, A. D. Chhokra, Z. Kang, H. Sun, A. Gokhale, and G. Karsai, "BARISTA: Efficient and scalable serverless serving system for deep learning prediction services," in *Proc. IEEE Int. Conf. Cloud Eng. (ICE)*, Jun. 2019, pp. 23–33.

[25] D. Damkevala, R. Lunavara, M. Kosamkar, and S. Jayachandran, "Behavior analysis using serverless machine learning," in *Proc. 6th Int. Conf. Comput. Sustain. Global Develop. (INDIACom)*, 2019, pp. 1068–1072.

[26] J. Carreira, P. Fonseca, A. Tumanov, A. Zhang, and R. Katz, "Cirrus: A serverless framework for end-to-end ML workflows," in *Proc. ACM Symp. Cloud Comput.*, 2019, pp. 13–24.

[27] H. Wang, D. Niu, and B. Li, "Distributed machine learning with a serverless architecture," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 1288–1296.

[28] M. Fotouhi, D. Chen, and W. J. Lloyd, "Function-as-a-service application service composition: Implications for a natural language processing application," in *Proc. 5th Int. Workshop Serverless Comput.*, 2019, pp. 49–54.

[29] C. Zhang, M. Yu, W. Wang, and F. Yan, "MArk: Exploiting cloud services for cost-effective, SLO-aware machine learning inference serving," in *Proc. USENIX Annu. Tech. Conf.* Renton, WA, USA: USENIX Association, Jul. 2019, pp. 1049–1062. [Online]. Available: https://www.usenix.org/conference/atc19/presentation/zhang-chengliang

[30] D. Barcelona-Pons, M. Sánchez-Artigas, G. París, P. Sutra, and P. García-López, "On the FaaS track: Building stateful distributed applications with serverless architectures," in *Proc. 20th Int. Middleware Conf.*, Dec. 2019, pp. 41–54.

[31] M. Zhang, C. Krintz, M. Mock, and R. Wolski, "Seneca: Fast and low cost hyperparameter search for machine learning models," in *Proc. IEEE 12th Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2019, pp. 404–408.

[32] A. Christidis, R. Davies, and S. Moschoyiannis, "Serving machine learning workloads in resource constrained environments: A serverless deployment example," in *Proc. IEEE 12th Conf. Service-Oriented Comput. Appl. (SOCA)*, Nov. 2019, pp. 55–63.

[33] A. Bhattacharjee, Y. Barve, S. Khare, S. Bao, A. Gokhale, and T. Damiano, "Stratum: A serverless framework for the lifecycle management of machine learning-based data analytics tasks," in *Proc. USENIX Conf. Oper. Mach. Learn. (OpML)*. Santa Clara, CA, USA: USENIX Association, May 2019, pp. 59–61. [Online]. Available: https://www.usenix.org/conference/opml19/presentation/bhattacharjee

[34] T. Rausch, W. Hummer, V. Muthusamy, A. Rashed, and S. Dustdar, "Towards a serverless platform for edge AI," in *Proc. 2nd USENIX Workshop Hot Topics Edge Comput. (HotEdge)*. Renton, WA, USA: USENIX Association, Jul. 2019, pp. 1–7. [Online]. Available: https://www.usenix.org/conference/hotedge19/presentation/rausch

[35] A. Dakkak, C. Li, S. G. D. Gonzalo, J. Xiong, and W.-M. Hwu, "TrIMS: Transparent and isolated model sharing for low latency deep learning inference in function-as-a-service," in *Proc. IEEE 12th Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2019, pp. 372–382, doi: 10.1109/CLOUD.2019.00067.

[36] Á. L. García, J. M. De Lucas, M. Antonacci, W. Zu Castell, and M. David, "A cloud-based framework for machine learning workloads and applications," *IEEE Access*, vol. 8, pp. 18681–18692, 2020.

[37] A. Kaplunovich and Y. Yesha, "Automatic tuning of hyperparameters for neural networks in serverless cloud," in *Proc. IEEE Int. Conf. Big Data*, Dec. 2020, pp. 2751–2756.

[38] A. Ali, R. Pinciroli, F. Yan, and E. Smirni, "BATCH: Machine learning inference serving on serverless platforms with adaptive batching," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, Nov. 2020, pp. 1–15.

[39] U. Elordi, L. Unzueta, J. Goenetxea, S. Sanchez-Carballido, I. Arganda-Carreras, and O. Otaegui, "Benchmarking deep neural network inference performance on serverless environments with MLPerf," *IEEE Softw.*, vol. 38, no. 1, pp. 81–87, Jan./Feb. 2021.

[40] C. Zhang, M. Yu, W. Wang, and F. Yan, "Enabling cost-effective, SLO-aware machine learning inference serving on public cloud," *IEEE Trans. Cloud Comput.*, vol. 10, no. 3, pp. 1765–1779, Sep. 2022.

[41] S. Shillaker and P. Pietzuch, "Faasm: Lightweight isolation for efficient stateful serverless computing," in *Proc. USENIX Annu. Tech. Conf.* Renton, WA, USA: USENIX Association, Jul. 2020, pp. 419–433. [Online]. Available: https://www.usenix.org/conference/atc20/presentation/shillaker

[42] J. R. Gunasekaran, C. S. Mishra, P. Thinakaran, M. T. Kandemir, and C. R. Das, "Implications of public cloud resource heterogeneity for inference serving," in *Proc. 6th Int. Workshop Serverless Comput.*, Dec. 2020, pp. 7–12.

[43] D. Chahal, R. Ojha, M. Ramesh, and R. Singhal, "Migrating large deep learning models to serverless architecture," in *Proc. IEEE Int. Symp. Softw. Rel. Eng. Workshops (ISSREW)*, Oct. 2020, pp. 111–116.

[44] J. Choi, J. Lee, and W. J. Cho, "Prognostics by classifying degradation stage on Lambda architecture," in *Proc. IEEE Int. Conf. Prognostics Health Manag. (ICPHM)*, Jun. 2020, pp. 1–9.

[45] M. Zhang, C. Krintz, and R. Wolski, "STOIC: Serverless teleoperable hybrid cloud for machine learning applications on edge device," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2020, pp. 1–6.

[46] M. Chadha, A. Jindal, and M. Gerndt, "Towards federated learning using FaaS fabric," in *Proc. 6th Int. Workshop Serverless Comput.*, Dec. 2020, pp. 49–54.

[47] D. M. Naranjo, S. Risco, G. Moltó, and I. Blanquer, "A serverless gateway for event-driven machine learning inference in multiple clouds," *Concurrency Comput., Pract. Exper.*, p. e6728, Dec. 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.6728, doi: 10.1002/cpe.6728.

[48] J. Jarachanthan, L. Chen, F. Xu, and B. Li, "AMPS-Inf: Automatic model partitioning for serverless inference with cost efficiency," in *Proc. 50th Int. Conf. Parallel Process.*, 2021, pp. 1–12.

[49] A. Kaplunovich and Y. Yesha, "Automatic hyperparameter optimization for arbitrary neural networks in serverless AWS cloud," in *Proc. 12th Int. Conf. Inf. Commun. Syst. (ICICS)*, May 2021, pp. 69–76.

[50] N. Shahidi, J. R. Gunasekaran, and M. T. Kandemir, "Cross-platform performance evaluation of stateful serverless workflows," in *Proc. IEEE Int. Symp. Workload Characterization (IISWC)*, Nov. 2021, pp. 63–73.

[51] M. S. Kurz, "Distributed double machine learning with a serverless architecture," in *Proc. Companion ACM/SPEC Int. Conf. Perform. Eng.*, Apr. 2021, pp. 27–33.

[52] J. Thorpe, Y. Qiao, J. Eyolfson, S. Teng, G. Hu, Z. Jia, J. Wei, K. Vora, R. Netravali, M. Kim, and G. H. Xu, "Dorylus: Affordable, scalable, and accurate GNN training with distributed CPU servers and serverless threads," in *Proc. 15th USENIX Symp. Operating Syst. Design Implement. (OSDI)*. Renton, WA, USA: USENIX Association, Jul. 2021, pp. 495–514. [Online]. Available: https://www.usenix.org/conference/osdi21/presentation/thorpe

[53] M. Zhang, C. Krintz, and R. Wolski, "Edge-adaptable serverless acceleration for machine learning Internet of Things applications," *Softw., Pract. Exper.*, vol. 51, no. 9, pp. 1852–1867, Sep. 2021.

[54] M. Sánchez-Artigas and P. G. Sarroca, "Experience paper: Towards enhancing cost efficiency in serverless machine learning training," in *Proc. 22nd Int. Middleware Conf.*, Dec. 2021, pp. 210–222.

[55] A. Grafberger, M. Chadha, A. Jindal, J. Gu, and M. Gerndt, "FedLess: Secure and scalable federated learning using serverless computing," 2021, *arXiv:2111.03396*.

[56] M. Yu, Z. Jiang, H. C. Ng, W. Wang, R. Chen, and B. Li, "Gillis: Serving large neural networks in serverless functions with automatic model partitioning," in *Proc. IEEE 41st Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2021, pp. 138–148.

[57] E. Paraskevoulakou and D. Kyriazis, "Leveraging the serverless paradigm for realizing machine learning pipelines across the edge-cloud continuum," in *Proc. 24th Conf. Innov. Clouds, Internet Netw. Workshops (ICIN)*, Mar. 2021, pp. 110–117.

[58] D. Chahal, M. Mishra, S. Palepu, and R. Singhal, "Performance and cost comparison of cloud services for deep learning workload," in *Proc. Companion ACM/SPEC Int. Conf. Perform. Eng.*, Apr. 2021, pp. 49–55.

[59] D. Chahal, S. Palepu, M. Mishra, and R. Singhal, "SLA-aware workload scheduling using hybrid cloud services," in *Proc. 1st Workshop High Perform. Serverless Comput.*, 2021, pp. 1–4.

[60] P. Patros, J. Spillner, A. V. Papadopoulos, B. Varghese, O. Rana, and S. Dustdar, "Toward sustainable serverless computing," *IEEE Internet Comput.*, vol. 25, no. 6, pp. 42–50, Nov. 2021.

[61] J. Jiang, S. Gan, Y. Liu, F. Wang, G. Alonso, A. Klimovic, A. Singla, W. Wu, and C. Zhang, "Towards demystifying serverless machine learning training," in *Proc. Int. Conf. Manag. Data*, Jun. 2021, pp. 857–871.

[62] A. Nesen and B. Bhargava, "Towards situational awareness with multimodal streaming data fusion: Serverless computing approach," in *Proc. Int. Workshop Big Data Emergent Distrib. Environ.*, Jun. 2021, pp. 1–6.

[63] J. Tagliabue, "You do not need a bigger boat: Recommendations at reasonable scale in a (mostly) serverless and open stack," in *Proc. 15th ACM Conf. Recommender Syst.*, Sep. 2021, pp. 598–600.

[64] F. Xu, Y. Qin, L. Chen, Z. Zhou, and F. Liu, "λDNN: Achieving predictable distributed DNN training with serverless architectures," *IEEE Trans. Comput.*, vol. 71, no. 2, pp. 450–463, Feb. 2022.

[65] T. P. Bac, M. N. Tran, and Y. Kim, "Serverless computing approach for deploying machine learning applications in edge layer," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2022, pp. 396–401.

[66] D. Barcelona-Pons, P. Sutra, M. Sánchez-Artigas, G. París, and P. García-López, "Stateful serverless computing with crucial," *ACM Trans. Softw. Eng. Methodol.*, vol. 31, no. 3, pp. 1–38, Jul. 2022.

[67] Y. Yang, L. Zhao, Y. Li, H. Zhang, J. Li, M. Zhao, X. Chen, and K. Li, "INFless: A native serverless system for low-latency, high-throughput inference," in *Proc. 27th ACM Int. Conf. Architectural Support Program. Lang. Operating Syst.*, Feb. 2022, pp. 768–781.

[68] P. G. Sarroca and M. Sánchez-Artigas, "MLLess: Achieving cost efficiency in serverless machine learning training," 2022, *arXiv:2206.05786*.

[69] H. B. Hassan, S. A. Barakat, and Q. I. Sarhan, "Survey on serverless computing," *J. Cloud Comput.*, vol. 10, no. 1, pp. 1–29, 2021.

[70] D. A. Tamburri, "Sustainable MLOps: Trends and challenges," in *Proc. 22nd Int. Symp. Symbolic Numeric Algorithms Sci. Comput. (SYNASC)*, Sep. 2020, pp. 17–23.

[71] J. M. Hellerstein, J. Faleiro, J. E. Gonzalez, J. Schleier-Smith, V. Sreekanti, A. Tumanov, and C. Wu, "Serverless computing: One step forward, two steps back," 2018, *arXiv:1812.03651*.

[72] H. van Kemenade. (Jan. 2022). *Python-Pillow/Pillow: 9.0.0*. [Online]. Available: https://doi.org/10.5281/zenodo.5813885

[73] D. G. McNeely-White, J. R. Beveridge, and B. A. Draper, "Inception and ResNet: Same training, same features," in *Proc. Biologically Inspired Cognit. Archit. Meeting*. Cham, Switzerland: Springer, 2019, pp. 352–357.

[74] Amazon. (2021). *Aws lambda*. [Online]. Available: https://github.com/aws/aws-lambda-go

[75] I. E. Akkus. (2021). *Knix*. [Online]. Available: https://github.com/knix-microfunctions/knix

[76] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, and M. Kudlur, "Tensorflow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. design Implement. (OSDI)*, 2016, pp. 265–283.

[77] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," *HotCloud*, vol. 10, nos. 10–10, p. 95, Jun. 2010.

[78] *AWS Lambda Now Supports Up to 10 Gb of Memory and 6 VCPU Cores for Lambda Functions*. Accessed: Jan. 17, 2022. [Online]. Available: https://aws.amazon.com/fr/about-aws/whats-new/2020/12/aws-lambda-supports-10gb-memory-6-vcpu-cores-lambda-functions/

[79] J. R. Gunasekaran, P. Thinakaran, M. T. Kandemir, B. Urgaonkar, G. Kesidis, and C. Das, "Spock: Exploiting serverless functions for SLO and cost aware resource procurement in public cloud," in *Proc. IEEE 12th Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2019, pp. 199–208.

[80] *AWS Lambda VS EC2: Which to Use and When | CBT Nuggets*. Accessed: Jan. 26, 2022. [Online]. Available: https://www.cbtnuggets.com/blog/certifications/cloud/aws-lambda-vs-ec2-which-to-use-and-when

[81] L. Wang, M. Li, Y. Zhang, T. Ristenpart, and M. Swift, "Peeking behind the curtains of serverless platforms," in *Proc. USENIX Annu. Tech. Conf.*, 2018, pp. 133–146.

[82] D. Mvondo, M. Bacou, K. Nguetchouang, L. Ngale, S. Pouget, J. Kouam, R. Lachaize, J. Hwang, T. Wood, D. Hagimont, and N. De Palma, "OFC: An opportunistic caching system for FaaS platforms," in *Proc. 16th Eur. Conf. Comput. Syst.*, 2021, pp. 228–244.

[83] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Proc. Eur. Symp. Res. Comput. Secur.* Cham, Switzerland: Springer, 2020, pp. 480–501.

[84] D. Crankshaw, X. Wang, G. Zhou, M. J. Franklin, J. E. Gonzalez, and I. Stoica, "Clipper: A low-latency online prediction serving system," in *Proc. 14th USENIX Symp. Networked Syst. Design Implement. (NSDI)*, 2017, pp. 613–627.

[85] A. Fuerst and P. Sharma, "FaasCache: Keeping serverless computing alive with greedy-dual caching," in *Proc. 26th ACM Int. Conf. Architectural Support Program. Lang. Operating Syst.*, Apr. 2021, pp. 386–400.

[86] J. Li, S. G. Kulkarni, K. K. Ramakrishnan, and D. Li, "Analyzing open-source serverless platforms: Characteristics and performance," 2021, *arXiv:2106.03601*.

[87] *Serverless: Develop & Monitor Apps on Aws Lambda*. Accessed: Jul. 11, 2022. [Online]. Available: https://www.serverless.com/

[88] A. Gujarati, S. Elnikety, Y. He, K. S. McKinley, and B. B. Brandenburg, "Swayam: Distributed autoscaling to meet SLAs of machine learning inference services with resource efficiency," in *Proc. 18th ACM/IFIP/USENIX Middleware Conf.*, Dec. 2017, pp. 109–120.

[89] *What is Mlops? | Nvidia Blog*. Accessed: Mar. 09, 2022. [Online]. Available: https://blogs.nvidia.com/blog/2020/09/03/what-is-mlops/

[90] D. Khatri, S. K. Khatri, and D. Mishra, "Potential bottleneck and measuring performance of serverless computing: A literature study," in *Proc. 8th Int. Conf. Rel., Infocom Technol. Optim. (ICRITO)*, Jun. 2020, pp. 161–164.

[91] Y. Wu, T. T. A. Dinh, G. Hu, M. Zhang, Y. M. Chee, and B. C. Ooi, "Serverless data science—Are we there yet? A case study of model serving," 2021, *arXiv:2103.02958*.

[92] J. Scheuner and P. Leitner, "Function-as-a-service performance evaluation: A multivocal literature review," *J. Syst. Softw.*, vol. 170, Dec. 2020, Art. no. 110708.

[93] M. Sadaqat, R. Colomo-Palacios, K. Ricardo, and L. E. S. Knudsen, "Serverless computing: A multivocal literature review," *NOKOBIT–Norsk Konferanse Organisasjoners Bruk AV Informasjonsteknologi*, 2018. [Online]. Available: http://hdl.handle.net/11250/2577600

[94] D. Taibi, N. El Ioini, C. Pahl, and J. R. S. Niederkofler, "Serverless cloud computing (function-as-a-service) patterns: A multivocal literature review," in *Proc. 10th Int. Conf. Cloud Comput. Services Sci. (CLOSER)*, 2020, pp. 1–12.

[95] M. M. John, H. H. Olsson, and J. Bosch, "Architecting AI deployment: A systematic review of state-of-the-art and state-of-practice literature," in *Proc. Int. Conf. Softw. Bus.* Cham, Switzerland: Springer, 2020, pp. 14–29.

[96] A. B. Kolltveit and J. Li, "Operationalizing machine learning models—A systematic literature review," in *Proc. IEEE/ACM 1st Int. Workshop Softw. Eng. Responsible Artif. Intell. (SERAI)*, May 2022, pp. 1–8.

[97] F. Jauro, H. Chiroma, A. Y. Gital, M. Almutairi, S. M. Abdulhamid, and J. H. Abawajy, "Deep learning architectures in emerging cloud computing architectures: Recent development, challenges and next research trend," *Appl. Soft Comput.*, vol. 96, Nov. 2020, Art. no. 106582.

[98] C. P. Filho, E. Marques, V. Chang, L. D. Santos, F. Bernardini, P. F. Pires, L. Ochi, and F. C. Delicato, "A systematic literature review on distributed machine learning in edge computing," *Sensors*, vol. 22, no. 7, p. 2665, Mar. 2022.

[99] G. A. S. Cassel, V. F. Rodrigues, R. D. R. Righi, M. R. Bez, A. C. Nepomuceno, and C. A. D. Costa, "Serverless computing for Internet of Things: A systematic literature review," *Future Gener. Comput. Syst.*, vol. 128, pp. 299–316, Mar. 2022.

[100] *Scopus vs Web of Science*. Accessed: Jan. 25, 2022. [Online]. Available: https://www.internauka.org/en/blog/scopus-vs-web-of-science

[101] P. D. Francesco, I. Malavolta, and P. Lago, "Research on architecting microservices: Trends, focus, and potential for industrial adoption," in *Proc. IEEE Int. Conf. Softw. Archit. (ICSA)*, Apr. 2017, pp. 21–30.

AMINE BARRAK received the master's degree in occurrence of security vulnerability changes in software code from Polytechnique Montreal. He is currently pursuing the Ph.D. degree in software engineering with the University of Quebec at Chicoutimi. His doctoral research focuses on the suitability between serverless computing and machine learning pipelines. He has currently published two in major refereed international conferences. His research interests include cloud computing, MLOps, application of machine learning techniques, analysis of software repositories, data science, data analytics, and natural language processing. He was a recipient of the Best Student Paper Award from CASCON 2018.

FABIO PETRILLO received the Ph.D. degree in computer science from the Federal University of Rio Grande do Sul, Brazil, in 2016. He is an Associate Professor with the Department of Software Engineering Information Technology, École de Technologie Supérieure, Canada. He was a Postdoctoral Fellow at Concordia University, Canada. He has was a programmer, software architect, manager, and agile coach for more than 20 years, working on critical mission projects and guiding several teams. He has published several papers in international conferences and journals, including IEEE TRANSACTIONS ON SOFTWARE ENGINEERING (TSE), *European Master on Software Engineering* (EMSE), *Journal of Systems and Software* (JSS), *Information and Software Technology* (IST), IEEE SOFTWARE, QRS, ICPC, SAC, ICSOC, and VISSOFT. His research interests include empirical software engineering, software quality, debugging, service-oriented architecture, cloud computing, and agile methods. He has been recognized as a Pioneer and an international reference on Software Engineering for Computer Games. He is the Creator of Swarm Debugging—a new collaborative approach to support debugging activities. He has served on the program committees of several international conferences, including QRS, CHI, SIGCSE, ICPC, VISSOFT, and GAS. He has reviewed for top international journals, such as IEEE TRANSACTIONS ON SOFTWARE ENGINEERING (TSE), *ACM Transactions on Software Engineering and Methodology* (TOSEM), *Journal of Systems and Software* (JSS), *European Master on Software Engineering* (EMSE), and *Information and Software Technology* (IST).

FEHMI JAAFAR received the Ph.D. degree from the Department of Computer Science, Montreal University, Canada. He was a Researcher at the Computer Research Institute of Montreal, an Adjunct Professor at Concordia University of Edmonton, and a Postdoctoral Research Fellow at Queen's University and Polytechnique Montreal. He is currently is an Associate Professor with Quebec University at Chicoutimi and an Affiliate Professor with Laval University and Concordia University. His researches have been published in top venues in computer sciences, including the *Journal of Empirical Software Engineering* (EMSE) and the *Journal of Software: Evolution and Process* (JSEP). He established externally funded research programs in collaboration with Defence Canada, Safety Canada, NSERC, MITACS, etc. His research interests include the Internet of Things security and the application of machine learning techniques in cybersecurity.