Received May 20, 2022, accepted June 1, 2022, date of publication June 9, 2022, date of current version June 20, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3181730

MLOps: A Taxonomy and a Methodology

MATTEO TESTI[1,2], MATTEO BALLABIO[2], EMANUELE FRONTONI[3], (Member, IEEE), GIULIO IANNELLO[4], (Member, IEEE), SARA MOCCIA[5,6], PAOLO SODA[4,7], (Member, IEEE), AND GENNARO VESSIO[8], (Member, IEEE)

[1]Integrated Research Centre, Università Campus Bio-Medico di Roma, 00155 Rome, Italy
[2]DeepLearningItalia, 24129 Bergamo, Italy
[3]VRAI Laboratory, Department of Political Sciences, Communication and International Relations, Università degli Studi di Macerata, 62100 Macerata, Italy
[4]Department of Engineering, Unit of Computer Systems and Bioinformatics, Università Campus Bio-Medico di Roma, 00155 Rome, Italy
[5]The BioRobotics Institute, Scuola Superiore Sant'Anna, 56127 Pisa, Italy
[6]Department of Excellence in Robotics and AI, Scuola Superiore Sant'Anna, 56127 Pisa, Italy
[7]Department of Radiation Sciences, Radiation Physics, Biomedical Engineering, Umeå University, 90187 Umeå, Sweden
[8]Department of Computer Science, Università degli Studi di Bari Aldo Moro, 70121 Bari, Italy

Corresponding author: Matteo Testi (m.testi@deeplearningitalia.com)

ABSTRACT Over the past few decades, the substantial growth in enterprise-data availability and the advancements in Artificial Intelligence (AI) have allowed companies to solve real-world problems using Machine Learning (ML). ML Operations (MLOps) represents an effective strategy for bringing ML models from academic resources to useful tools for solving problems in the corporate world. The current literature on MLOps is still mostly disconnected and sporadic. In this work, we review the existing scientific literature and we propose a taxonomy for clustering research papers on MLOps. In addition, we present methodologies and operations aimed at defining an ML pipeline to simplify the release of ML applications in the industry. The pipeline is based on ten steps: business problem understanding, data acquisition, ML methodology, ML training & testing, continuous integration, continuous delivery, continuous training, continuous monitoring, explainability, and sustainability. The scientific and business interest and the impact of MLOps have grown significantly over the past years: the definition of a clear and standardized methodology for conducting MLOps projects is the main contribution of this paper.

INDEX TERMS MLOps, continuous monitoring, continuous integration, continuous delivery, continuous training, XAI, sustainability.

I. INTRODUCTION

In the last decades, Machine Learning (ML) has emerged as a powerful tool to solve complex real-world problems such as stock prediction [1], biomedical image analysis [2]–[4], autonomous driving [5], and fraud detection [6]. Since data availability has reached levels never seen before, businesses around the world are working to leverage these data and process them automatically, exploiting the generalization power of ML to take actions and decisions [7].

In most real-world applications, data are constantly changing. This implies that ML models need to be retrained or, in the worst-case scenario, the entire ML pipeline has to be rebuilt to tackle feature drift [8], [9]. A more frequent, faster, and simpler release cycle helps meet any regulatory or business changes. To achieve industrial growth, standardized production methods are required [10]–[12]. To industrialize ML models, a good set of production methods must be applied [13]. One of the key elements in facilitating the development of industry-leading companies is to improve communication between Science Technology Engineering Math (STEM) professionals and industry leaders or industry professionals by adopting a proven set of steps for industrializing ML solutions [14], [15].

Machine Learning Operations (MLOps) is a candidate to define these standardized production methods [16], [17]. MLOps can be viewed as the iterative process of pushing the latest best ML models to production [18], [19]. In fact, conducting an MLOps project means supporting automation, integration, and monitoring at all stages of building an ML system, including training, integration, testing, release, deployment, and infrastructure management [20], [21].

MLOps was born from different fields: ML, Development and Operations (DevOps), and data engineering (Fig. 1).

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

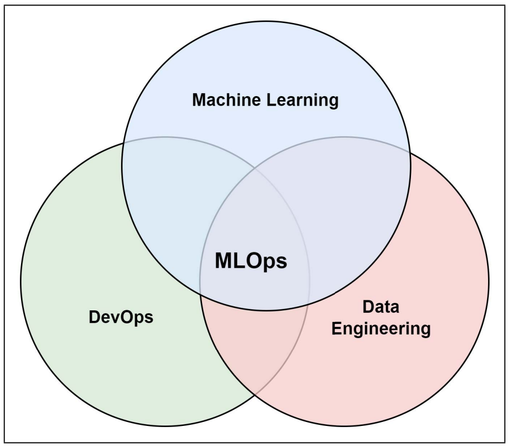


FIGURE 1. MLOps develops upon Machine Learning, DevOps, and data engineering.

Of the three fields, DevOps had the biggest impact on MLOps development. DevOps is a method of thought and practice that aims to improve and remove as much as possible the friction between development and operations (implementation and integration), seeing them as a single process [22]. The goal of DevOps is to study ways to improve service quality and features to meet customer needs [23], [24]. The primary links between MLOps and DevOps are the concepts of continuous integration (CI) and continuous delivery (CD), which allow software to be produced in short cycles, ensuring that it can be reliably released at any time.

When we look at how the current literature describes an ML project life-cycle, a picture like the one illustrated in Fig. 2 (top) is often shown. In many companies, model development and operations are carried out manually and without implementing MLOps. This slows down the industrialization of ML methodologies. As ML models have zero Return on Investment (ROI) until they can be used [25], [26], time to market should be the first metric to look at and optimize for any ML project. The only way to improve the release and continuous use of ML solutions in the industrial environment is to take great care of the part following the development of the model, in particular the interface between the ML solution and the existing Information and Communication Technologies (ICT) system. In fact, the most time-consuming step in releasing an ML solution into production is Operations (Fig. 2 (bottom)). It is worth noting that, although MLOps fosters process automation, its main goal is not to optimize business [27].

II. OBJECTIVES

The main objective of this paper is to provide a literature review on MLOps to highlight current challenges in building and maintaining an ML system in a production environment [28]. At the same time, we aim at giving an overview of why MLOps was introduced to translate ML systems into production [29], [30]. To this end, we selected papers and projects in the field of MLOps and propose a taxonomy to understand the work done so far. We identify key concepts by analyzing existing literature from 2015 to 2022. Finally, we propose our operational methodology to approach an ML project. As far as we know, this is the first effort to systematize the literature on this topic and provide its operationalization.

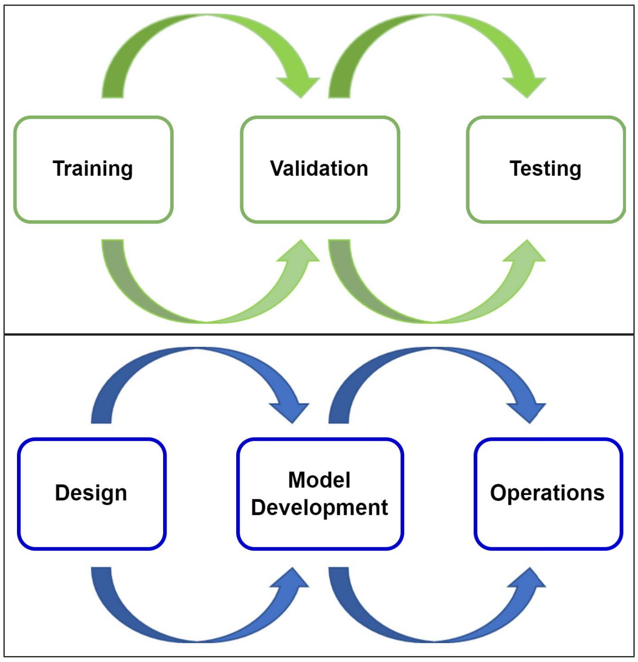


FIGURE 2. Traditional workflow of ML (top) vs. MLOps workflow (bottom).

The main difference between the operational methodology proposed in this paper and the traditional workflow implemented in many ML projects consists in the full integration of the various project steps to realize an effective, scalable, but above all industrializable solution. In fact, in most ML projects all forces are used for the development of an accurate ML model, without giving due importance to the integration and monitoring of the ML solution in the industrial environment.

The main motivation for proposing a methodology is to try to normalize each step to bring ML models from research to production. Due to the growing interest, researchers are trying to figure out each step of MLOps without involving business partners in defining each step. This results in a misalignment in the definition of the MLOps issues without having a clear vision from the transition from research to production up to the maintenance of the models. By following a clear methodology, teams can have a deeper overview of all processes and organize each part of a project in a better and more systematic way.

The rest of this paper is structured as follows: Section III reviews the related literature; Section IV presents the proposed MLOps workflow; Section V concludes the paper and suggests high-level directions for further research.

III. PROPOSED TAXONOMY

As introduced in Section I, MLOps initiatives aim to establish resilient and efficient workflows by creating robust pipelines [31], established practices [32], and auxiliary frameworks and tools. Indeed, model development is only a small part of the overall process, and many other processes, configurations, and tools need to be integrated into the system [33]. Bringing the application of DevOps techniques in the context of continuous training (CT), CD, CI, and continuous monitoring (CM) is among the main requirements of an ML project that aims to provide process automation, governance, and agility.

In the literature, several projects have tried to tackle various aspects of the ML production process by expanding existing libraries or by creating new tools to enhance the quality and performance of specific processes or make them more insightful. Up to now, there is no standardized and common pipeline to follow for an end-to-end MLOps project. To cluster the different approaches, we propose the following taxonomy:

1) *ML-based software systems*, also known as *model-centric frameworks*. These systems focus on the architecture of ML models with a view to (CI/CD) [23], [34], [35]. The goal of such systems is twofold: on the one hand, it is to create and automate ML pipelines; on the other hand, the goal is to increase the level of automation in the ML software life-cycle [36].
2) *ML use case applications* where, for example, papers explain an MLOps workflow to foster collaboration and negotiation between surgeon and patient [37], [38] or the ML pipeline on the Cloud for drug discovery [39].
3) *ML automation frameworks* such as MLFlow [40], Kedro [41] or Amazon SageMaker [42], and *benchmarking frameworks* such as MLPerf [43], MLModelScope [44] and Deep500 [45]. These are interesting commercial tools that are already being used in daily work practice and represent excellent ML framework automation solutions.

The following subsections review in more detail the works that fall into the three categories.

A. ML-BASED SOFTWARE SYSTEMS

Machine Learning is becoming the primary approach to solving real-world problems. Therefore, there are many data science teams studying how to apply DevOps principles to industries. The ML life-cycle involves manual steps for deploying the ML pipeline model. This method can produce unexpected results due to the dependency on data, preprocessing, model training, validation, and testing. The idea is to design an automated pipeline using two DevOps principles which are CI and CD. The functionality of CI is to test and validate data, data schemes, and models. CD is for an ML pipeline that should automatically deploy another ML service [23]. The ML life-cycle has different methodologies to fit different scenarios and data types. The approach most used by data mining experts is CRoss-Industry Standard Process for Data Mining (CRISP-DM) [46], introduced in 1996 by Daimler Chrysler. Experts can borrow the standard CRISP-DM methodologies and try to apply them to the MLOps pipeline. The process typically involves two teams: ML scientists responsible for model training and testing, and ML engineers responsible for production and deployment. MLOps pipeline automation with CI/CD routines is as follows:

- Business problem analysis;
- Dataset features and storage;
- ML analytical methodology;
- Pipeline CI components;
- Pipeline CD components;
- Automated ML triggering;
- Model registry storage;
- Monitoring and performance;
- Production ML service.

One of the points of greatest attention after CI and CD is monitoring, in terms of metrics and Key Performance Indicators (KPIs), and the continuous deployment of models. This part includes model performance, data monitoring, outlier detection, and explanations of historical predictions. Continuous monitoring is a process that allows understanding in real-time when validation performance tends to decrease. Outlier detection is the key to trusting and keeping the model healthy. Therefore, the most important function of continuous monitoring is to ensure high model performance and KPIs used to validate models. There are many metrics to test the quality of a model, such as precision, recall, F1, and MSE. However, these metrics evaluate a model in the laboratory, regardless of the real-world context of how the model will be used. When evaluating ML models in the context of real applications, model performance metrics are not enough to establish the robustness of the models. The most basic step towards supporting such KPI-based analytics is to ensure that KPIs and model metrics are stored with a common correlation ID to identify which model operations contributed to transactions with a particular KPI score [36]. Other important KPIs at the company level for evaluating the performance of the model can be: time-to-market, infrastructure cost, scalability, and profitability indices on sales (ROS) [47]. Unfortunately, ML models often fail to generalize outside the training data distribution [48].

Finally, the trust in the ML project is the model explanation. Explainability allows users to trust the prediction and this improves transparency. The user can verify which factors contributed to certain predictions, introducing a layer of accountability [35]. The terms "explainability" and "interpretability" are being used interchangeably throughout the literature; however, in the case of an AI-based system, explainability is more than interpretability in terms of importance, completeness, and fidelity of predictions or classifications [49]. Explainable Artificial Intelligence (XAI) is a research trend that promotes explainable decision-making. Many real-world ML applications greatly increase the efficiency of industrial production from automated equipment and production processes [50]. However, the use of "black-boxes" has not yet been overcome due to the lack of explainability and transparency of the models and decisions still present [51].

B. ML USE CASE APPLICATIONS

One of the most difficult challenges is using ML in real-world applications where the focus is on system integration and scaling. The setup of MLOps use cases is continuous training, continuous integration, and continuous deployment [52], where new versions of the ML system can be deployed in running software. In this section, we present a case study to understand what a workflow looks like in an MLOps project. The use case concerns Oravizio [38], a software product that provides data-driven information on patient-level risks related to hip and knee joint replacement surgery. Oravizio helps the collaboration and negotiation between the surgeon and a patient so that the decisions that are taken are informed and there is consent to the operation.

Oravizio provides three different dedicated prediction models:

- Risk of infection within one year from surgery;
- Risk of revision within two years from surgery;
- Risk of death within two years from surgery.

In the case of Oravizio, data were collected over the years, including 30, 000 medical records, from patients who have undergone surgery. Since the number of cases is so large that no surgeon can process them manually during the appointment, these data have been used to create a risk calculation model that predicts the outcome of the surgery. The various formats of the data were one of the issues during pre-processing to create a standard for later analysis [37]. Once the data are standardized, an ML model can be created for each risk to enable validation and ensure regulatory compliance. The models selected to be trained for this task were Logistic regression, Random forest, XGBoost, and Weibull/Cox survival mode. According to the results, gradient boosting with XGBoost produced the best performance and can be selected for use in production [38].

As shown in Fig. 3, these models are usually re-trained during the life-cycle of an ML product. We have new data and this entails continuous training to improve accuracy. We also have continuous delivery in terms of deploying new models and continuous monitoring, which has two faces: some indexes to track accuracy for data science analysis, and some KPI or different indexes from the business or clinical side to help understand the model and whether this approach can improve the business.

Unfortunately, there are no other use cases available in the literature that have a clear pipeline of MLOps where it is clearly explained the process from problem understanding to model deployment and continuous training, delivery, and monitoring. For example, in the case of the Uffizi Gallery in Florence [34], one of the most visited museums in Italy with over 2 million visitors, the project aims to reduce the queue using ML but we do not have a clear set-up of the MLOps workflow. In the article in question, the authors talk about the chosen architecture, the reason why it was decided to use

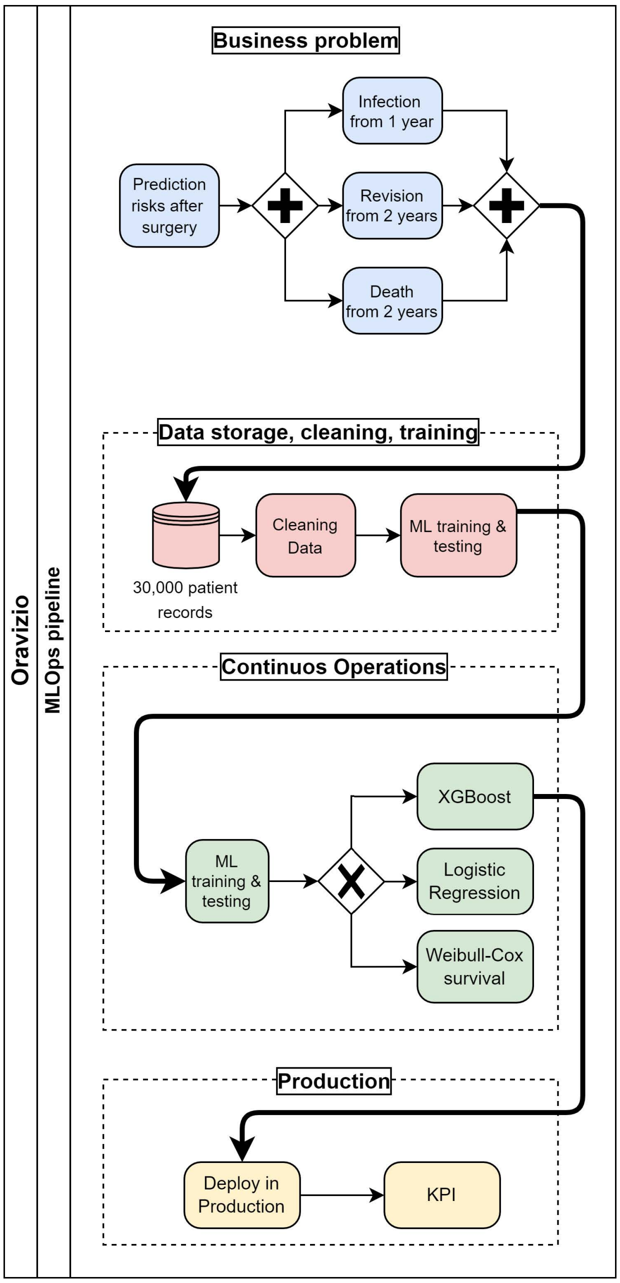


FIGURE 3. Workflow of Oravizio.

an ML algorithm and the run-time continuous training of the algorithm to improve performance, but what is missing is a methodological guideline.

C. ML AUTOMATION FRAMEWORKS

To have a business impact, ML applications need to be deployed in production, which means deploying a model in a way that can be used for inference (e.g., REpresentational state transfer (REST) services) and deploying scheduled jobs to update the model regularly. This is especially challenging when deployment requires collaboration with another team, such as application engineers who are not ML experts, or when the ML team uses different libraries or

TABLE 1. Some popular data labeling tools.

Name	Status	Launched in
Doccane	Open source	2019
Amazon SM	Private	2017
Labelbox	Private	2017
V7 Darwin	Private	2018
Dataloop	Private	2017
SuperAnnotate	Private	2018
Supervisely	Private	2017

TABLE 2. Pros and cons of some popular data labeling tools.

	Amazon SM	SuperAnnotate	V7
Free trial	✗	✓	✓
Automatic labeling	✓	✓	✓
Document processing	✗	✓	✓
Image processing	✓	✓	✓
Team collaboration	✓	✗	✓
Pros	Control of ML models with sagemaker endpoints	Image/video annotation (useful for Computer Vision workflows)	The API and CLI tools allow to work flexibly
Cons	Pricing is high when running high amount of labels	Team collaboration is not possible	Reporting and analytics integration are not possible

frameworks [40]. ML projects have created new challenges that are not present in traditional software development. One of these includes tracking input data, data versions, tuning parameters, and so on, to keep production deployment up-to-date [53]. In this section, we want to summarize these challenges and describe some of the most popular ML frameworks like MLflow, Kubeflow, MLPerf, etc. [54].

MLOps frameworks can be divided into three main areas [55] dealing with:

- Data management;
- Modelling;
- Operationalization.

1) DATA MANAGEMENT

Data labeling tools (Table 1) are used to help the data science team to label large datasets such as texts, images, etc. [56], [57]. Labeled data are used to train supervised ML algorithms. We provide an overview of some data labeling tools and advantages and disadvantages in Table 2.

Data versioning tools (Table 3), on the other hand, are used by data science and data engineering teams to manage different versions of models and datasets [58]. This helps data science teams gain insights, such as identifying how data changes impact model performance and understanding how datasets evolve. An overview of some popular data versioning tools along with pros and cons are shown in Table 4.

TABLE 3. Some popular data versioning tools.

Name	Status	Launched in
Comet	Private	2017
DVC	Open source	2017
Delta Lake	Open source	2019
Dolt	Open source	2020
LakeFS	Open source	2020
Pachyderm	Private	2014
Qri	Open source	2018
Weight Biases	Private	2018

TABLE 4. Pros and cons of some popular data versioning tools.

	DVC	Delta Lake	Dolt
Data format agnostic	✓	✗	✗
Cloud storage	✓	✓	✗
Simple to use	✓	✗	✗
Support for Big Data	✗	✓	✗
Pros	Lightweight and usable with all Cloud platforms and storage types	Offers many features such as ACID transactions or metadata management	Similar to SQL (more accessible and less complex)
Cons	DVC version control is too coupled with pipeline management (redundancy)	Requires using a dedicated data format (less flexibility)	Dolt is a DB and is still a maturing product

TABLE 5. Some popular feature engineering tools.

Name	Status	Launched in
AutoFeat	Open source	2019
dotData	Private	2018
Feast	Open source	2019
FeatureTools	Open source	2017
Rasgo	Private	2020
TSFresh	Open source	2016

2) MODELLING

In Table 5 and 6, we present feature engineering tools that allow adding automation to the process of extracting useful features from raw datasets to create better training data [59]. These tools help speed up the process of feature engineering and extraction and create better training data for ML models.

Developing ML projects involves running multiple experiments with different models, model parameters, or training data. Experiment tracking tools save all necessary information about different experiments [60]. This allows to track the versions of experiment components and results and allows for comparison between different experiments. Some examples of experiment tracking tools are shown in Table 7. In Table 8, a summary of their pros and cons is presented.

Hyperparameters are the main part to get better models. These are the parameters of the ML training algorithms such as the learning rate, the type of regularization applied, and so

TABLE 6. Pros and cons of some popular feature engineering tools.

	AutoFeat	FeatureTools	TsFresh
Pros	Can Easily handle categorical features with one-hot encoding and feature selection	Custom transformations applied to data. Best at handling relational databases	Best open source Python tool available for time series classification and regression
Cons	Not good at handling relational data. Just make simple features and basic transformations	For databases that are not relational, normalization is necessary. Does not have support for unstructured data	Can only be used for time series data (less flexibility)

TABLE 7. Some popular experiment tracking tools.

Name	Status	Launched in
Comet	Private	2017
Guild AI	Open source	2019
ModelDB	Open source	2020
Neptune.ai	Private	2017
TensorBoard	Open source	2017
Weight Biases	Private	2018

TABLE 8. Pros and cons of some popular experiment tracking tools.

	TensorBoard	Neptune.ai	Comet
Open source	✓	✗	✗
Platform and language agnostic	✗	✗	✗
Easy to set-up	✓	✓	✓
Scalable for many experiments	✗	✓	✓
Pros	Large library of pre-built tracking tools and easy integration	Flexible and works well with other frameworks	Features for sharing work and data science workflows
Cons	May not scale well with large experiments	Paid solutions may be expensive	The free tier only offers one free private project

on. Hyperparameter tuning tools help automate the process of searching and selecting the optimal hyperparameters that perform better [61], [62]. Popular hyperparameter tuning tools are shown in Table 9 and 10.

3) OPERATIONALIZATION

ML model deployment tools facilitate the integration and deployment of ML models into production [63]. Some tools with advantages and disadvantages for each software are shown in Table 11 and Table 12.

ML model monitoring is another important part of a successful ML project because ML model performance tends to decay after model deployment due to changes in the input data stream over time [64], [65]. Model monitoring tools detect data drift and anomalies over time and allow to set alerts in case of performance issues. An overview of some popular data monitoring tools is provided in Table 13 and in Table 14, with advantages and disadvantages.

There are also tools that cover the end-to-end ML lifecycle [66]. Some popular platforms are shown in Table 15 and in Table 16 with advantages and disadvantages.

TABLE 9. Some popular hyperparameter tuning tools.

Name	Status	Launched in
Google Vizier	Public	2017
Hyperopt	Open source	2013
Optuna	Open source	2018
Scikit-Optimize	Open source	2016
SigOpt	Public	2014
Talos	Open source	2018

TABLE 10. Pros and cons of some popular hyperparameter tuning tools.

	HyperOpt	Optuna	SigOpt
Pros	Designed to accommodate Bayesian optimisation algorithms	User API to dynamically build search spaces for hyperparameters	Black-box hyperparameter optimisation solution that automates model tuning
Cons	Missing some documentation	Pruning in Optuna automatically stops unpromising trials early during training (can be a problem)	Black-box optimization can be a problem (less flexibility)

TABLE 11. Some popular model deployment tools.

Name	Status	Launched in
Algorithmia	Private	2014
BentoML	Open source	2019
Kubeflow	Open source	2018
Seldon	Private	2020
Tensorflow Serving	Open source	2016
Torch Serve	Open source	2020

IV. PROPOSED MACHINE LEARNING OPERATIONS METHODOLOGIES

In this section, we provide our methodology for an MLOps project that aims to unify the lessons learned from the literature review into a single framework. The main difference from the other frameworks is that we are trying to create a new standard for ML projects inspired by CRISP-DM that helps strengthen the link between research and industries. Below, the different stages of the proposed MLOps process are described. Figure 4 provides a schematic overview.

A. BUSINESS PROBLEM UNDERSTANDING

Establishing a business understanding and the success criteria for solving the problem under study is the first step in an ML project [67]. Business understanding is a non-technical

TABLE 12. Pros and cons of some popular model deployment tools.

	TensorFlow Serving	Kubeflow	Seldon
Pros	Many batch requests to the same model, model versioning management	Offers monitoring and status control and simplifies the onboarding of new team members	Custom offline models. Real-time predictions that expose APIs to external clients
Cons	Works only with TensorFlow models	Difficult to set-up and configure manually. High availability is not automatic (manual configuration)	Set-up can be a bit complex

TABLE 13. Some popular model monitoring tools.

Name	Status	Launched in
Arize	Private	2020
Evidently AI	Open source	2020
Fiddler	Private	2018
Losswise	Private	2018
Neptune.ai	Private	2017
Unravel Data	Private	2013

phase and, for this reason, communication between data scientists and business experts is the main part of identifying the business problem. During this phase, it is essential to map the processes, systems, key data elements, and policy documentation for the key domains expressed in the business problem. This information is often created and maintained by the data governance team with an enterprise data governance. The initial step is gathering requirements and clearly defining the objectives and key results (OKR). In this part, data scientists should discuss with business experts to determine if ML can really help. For each of the OKRs, it is necessary to define one or more KPIs [68]. These KPIs need to be documented for future reference and will be critically useful in ensuring that the project delivers the expected value. The KPIs must match the metrics (MSE, accuracy, etc.) used by the data science team to understand how model improvement impacts the business. The definition and documentation of business problems provide a key context for the subsequent phases, helping to distinguish relevant data, defining how data maps into the model (both during training and deployment), and identifying which dimensions of the model performance should be monitored once the model is in production and according to what criteria [69].

B. DATA ACQUISITION

During data acquisition, the goal is simply to collect enough data to train the ML model to get the first solution [70]. The data scientist identifies information in terms of features/attributes presented for a specific business problem. These aspects should be discussed with a field-expert data engineer to identify potential data sources. Once the dataset

TABLE 14. Pros and cons of some popular model monitoring tools.

	Neptune.ai	Arize	Evidently AI
Pros	Flexible metadata structure, customization of dashboards and log performance metrics	Simple integration, pre-launch validation and automatic monitoring	Open-source ML model monitoring system and integration with Pandas DataFrames
Cons	Paid solutions might be expensive	Paid solutions might be expensive	Too constrained to use Pandas DataFrames

TABLE 15. Some popular ML life-cycle tools.

Name	Status	Launched in
Alibaba ML	Public	2018
Amazon SM	Public	2017
Cloudera	Public	2020
Databricks	Private	2015
DataRobot	Private	2019
Google Cloud	Public	2008
H2O.ai	Open source	2012
Iguazio	Private	2014
Microsoft Azure	Public	2010
MLFlow	Open source	2018

TABLE 16. Pros and cons of some popular ML life-cycle tools.

	Amazon SM	Microsoft Azure	Google Cloud
Pros	Many features that make building, training, monitoring and deploying ML models easy	Azure ML platform supports Python, R, Jupyter and RStudio with automated ML	Possibility to perform tasks without writing any code using the Auto ML feature with an easy-to-use UI
Cons	Paid solutions (first two months free)	Paid solutions, but 12-month free service and credits to explore Azure	Less customization because codeless

is identified, the data engineer builds the pipeline that makes the data available to the data scientist. The data engineer performs the preliminary cleaning and validation steps so that there is a sufficient amount of high-quality data to meet the data scientist's needs.

The tasks for data acquisition can be summarized as follows:

- Data Extraction: select and integrate the data relevant to the ML task.
- Data Analysis: exploratory data analysis to understand the data schema and the characteristics expected by the model.
- Data Preparation: identify the data preparation and feature engineering required for the model. This preparation involves data cleaning and splitting into training, validation, and test set. Data transformation and feature engineering also apply to the model that solves the target task. The output of this step is data ready in the prepared

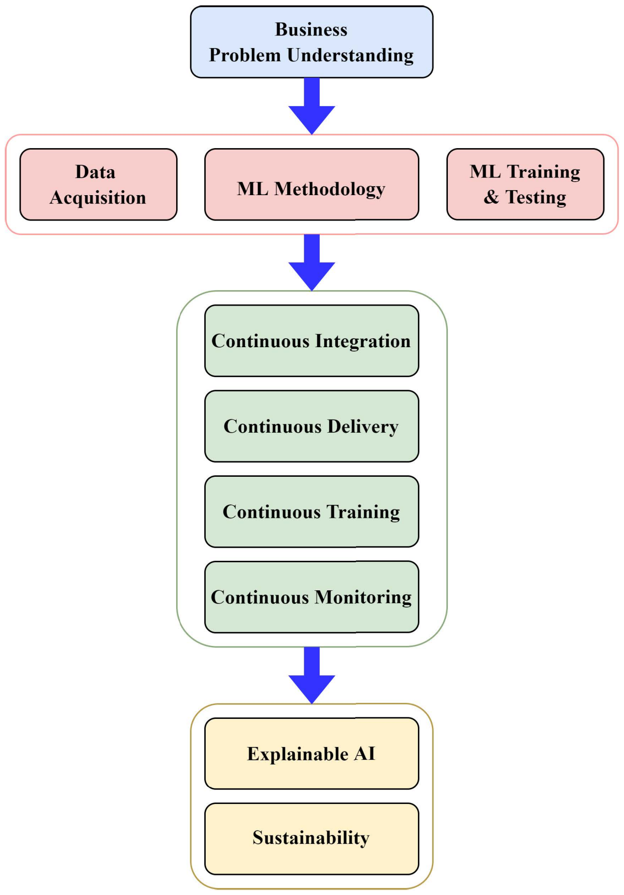


FIGURE 4. **Proposed MLOps workflow.**

format. For example, NULL values are converted to zero, or outliers are excluded from the dataset.

When there is not enough data to train the model, two main methodologies allow to bypass the problem:

- Data Augmentation is a technique that allows increasing the number of data available by inserting copies of the data (e.g., in the case of images, we use the same rotated, enlarged, blurred, etc.).
- Transfer Learning, which allows reusing most of the weights of a neural network already trained on a similar problem.

C. ML METHODOLOGY

After data acquisition, selecting the best ML algorithms to solve the problem is a key part of the ML project. Usually, the data science team studies the state-of-the-art for the specific problem and tries a bottom-up approach to solving it. ML is experimental by nature, trying different features, models, parameters and hyperparameter configurations to find what works best. The bottom-up approach typically consists in trying different models with increasing degrees of complexity until reaching the best one. This methodology helps data scientists to start with simple models before trying to implement complex ones.

D. ML TRAINING AND TESTING

The process of training and optimizing a new ML model is an iterative process in which data scientists test several algorithms, features, and hyperparameters. Once the best ML models have been chosen, they are re-trained and tested. The models are evaluated using different validation methods such as:

- Holdout validation, this is a type of external validation in which the dataset is split into two randomly sized subgroups.
- Cross-validation, in which the original sample is randomly partitioned into k equal-sized subgroups. Of the k subgroups, one subsample is kept as a testing dataset and $k-1$ as training.
- Bootstrap validation, in which we resample the dataset with replacement producing new datasets with the same number of instances as the initial dataset.

The output of this step is a set of metrics for evaluating the quality of the model. Once this iteration is complete, the weights of the best models are saved and deployed using an API infrastructure. Training and testing an ML system is integration, data validation, trained model quality evaluation, and model validation. The main goal is to keep track of all experiments and maintain reproducibility while maximizing code reusability [71]. We have seen that there exist different tracking tools which can simplify the process of storing the data, the features selected, and model parameters along with performance metrics. These allow to compare the differences in performance and aid the reproducibility of the experiments. Without reproducibility, data scientists are unable to deliver the model to DevOps to see if what was created in the lab can be faithfully reproduced in production [72].

E. CONTINUOUS INTEGRATION

Continuous integration is a well-established development practice in the software development industry [52] and is the first step in starting the continuous delivery journey. CI enables companies to have frequent releases, and improve software quality and teams' productivity [73]. This practice includes automated software building and testing [74].

In the continuous integration pipeline, we build source code and run various ML trained models. The outputs of this stage are components (packages and artifacts) to be deployed in the pre-production/production environment of continuous delivery [75]. The ML code is a small portion of a real ML system because an important component is the infrastructure, configuration, and data elaboration. Continuous integration for ML systems relies on having a substantial impact on the end-to-end pipeline to automate the delivery of the ML models with minimal effort. The main steps for continuous integration are [22]:

- Source code management (SCM);

- Push/pull changes to the repository to trigger a continuous delivery build;
- Check the latest code and associated data version from the data repository storage;
- Running of the unit tests;
- Building/running of the ML model;
- Testing and validation;
- Packaging of the model and building of the container image;
- Pushing of the container image to the registry.

Several software tools have been used for the deployment of ML models such as Jenkins [76], Git [77], Docker [78], Helm [79], and Kubernetes [80]. Then, to summarize, the pipeline and its components are built, tested, and packaged when new code is committed or pushed to the source code repository. CI is testing and validating code, dataset, data schemas, and models. The validated model is deployed to a target environment to provide predictions. This deployment can be one of the following:

- Microservices with a REST API to provide online predictions;
- A model embedded into an edge or mobile device;
- Part of a batch prediction system.

F. CONTINUOUS DELIVERY

Continuous delivery has the goal to ensure that an application is always in a production-ready state after successfully passing the automated tests and quality checks [81]. The object of the deployment stage is to enable a seamless roll-out of new models, with the lowest possible risk. Best practices in the continuous delivery of software services involve the use of safe deployment techniques, such as A/B tests. CD is an ML pipeline that should automatically deploy model services. CD employs a set of practices such as CI, and deployment automation to automatically deliver software in production [82]. CD is a push-based approach [83] and this practice has reduced deployment risk, lowered costs, and gained user feedback faster.

In this phase, the construction of artifacts takes place, which were produced by previous continuous integration in the staging/pre-production/production environment. Test models are obtained from this phase. The components of the CD pipeline are summarized as follows:

- Staging environment: deploying the trained ML model first in a staging environment is a standard operation in ICT. The output of this step is a test model that is pushed into the model registry archive.
- Model register archiving: necessary to define an archiving location where ML models in staging state and ML models in production state are loaded.
- Automatic activation: this step is performed automatically according to a schedule or a response in the production environment. The output of this phase is a test model that is pushed into the staging environment.

G. CONTINUOUS TRAINING

During continuous training, we need to keep storing more data and setting up the data in the same way we train our model. This means detecting outliers to understand when the data distribution diverges from the training data. CT is concerned with automatically retraining and serving models [84]. Continuous training is a part of MLOps which automatically and continuously retrains models before they are redeployed.

To design a continuous training strategy, we should answer the following questions [85]:

- *When should a model be retrained?*
  - Periodic training.
  - Performance-based trigger.
  - Trigger based on data changes.
  - Retraining on demand.
- *How much data is needed for retraining?*
  - Fixed window.
  - Dynamic window.
  - Representative subsample selection.
- *What should be retrained?*
  - Continual learning vs. transfer learning.
  - Offline (batch) vs. online (incremental) learning.
- *When to deploy the model after retraining?*
  - A/B testing.

H. CONTINUOUS MONITORING

The main objective during the monitoring stage is to manage the risks of the in-production models by checking for performance drift [86] and alerting an operator that model accuracy has dropped. The model predictive performance is monitored to potentially invoke a new iteration in the ML process. Once the model has been deployed to production, it still needs continuous validation or testing because patterns in the data can change over time. The model may become less accurate because the data used in training the model are no longer representative of the new data existing in production [71]. Performance monitoring not only affects the quantitative performance metrics. Therefore, during the continuous monitoring, both metrics and the KPIs from the technical part to the business part must be taken under control.

I. EXPLAINABLE AI

Deep Learning methods [87] now dominate benchmarks on different tasks and achieve superhuman results. This improvement has often been achieved through increased model complexity. Once these models have become a real application in production, the community has started studying the "explainability" of the models to answer business questions. Explainability can be defined as "the degree to which a human can understand the cause of a decision" [88]. Explainability is mostly connected with the intuition behind the outputs of a model [89]; therefore, an ML system is explainable when it is easier to identify cause-and-effect relationships within the system inputs and outputs. For example, in image recognition tasks, part of the reason that led a system to decide that a specific object is part of an image (output) could be certain dominant patterns in the image (input). The more explainable a model is, the greater the understanding practitioners get in terms of internal business procedures that take place while the model is making decisions. An explainable model does not necessarily translate into one that humans can understand (internal logic or underlying processes) [90]. The explainability of the model allows the user to build trust in the predictions made by the deployed system and improve transparency. The user can verify which factors contributed to certain predictions, introducing a layer of accountability [15].

J. SUSTAINABILITY: CARBON FOOTPRINT

The increasingly common use of Deep Learning models in real-world projects, as the other side of the medal, corresponded to immense growth in the computation and energy required [91]. If this growing trend continues, Deep Learning could become a significant contributor to climate change. This trend can be mitigated by exploring how to improve energy efficiency in the DL models [92]. Hence, data scientists need to know their energy and carbon footprint, so that they can actively take steps to reduce them whenever possible. Carbon footprint is a measure of the total exclusive amount of carbon dioxide emissions that are directly and indirectly caused by an activity or accumulated during the life stages of a product [93].

Strubell *et al.* selectively focused on carbon footprint analysis on AI models for natural language processing [94]. For example, the training of an NLP Transformer model was estimated to be equivalent to that of a commercial flight between San Francisco and New York. The publication of these estimates has had a significant effect in the scientific world. Following the publication of these data, the 2020 White Paper on AI released by the European Commission has called for actions that go beyond the collection of impressive but admittedly anecdotal data about the training of selected AI systems [95]. For this reason, it is necessary to calculate the carbon footprint of each individual AI system and the AI sector [96].

It is important to emphasize that, during the MLOps lifecycle, carbon footprint should be taken into account when choosing models. It should be better to take a bottom-up approach trying the first simple models without jumping to the state-of-the-art with complex and expensive models. The same approach is to calculate the carbon footprint during training and testing, but also during continuous integration, continuous delivery, and continuous training.

V. CONCLUSION

In this paper, we have provided an overview of approaches in the literature using MLOps: we have provided a taxonomy of the current literature and proposed a methodology for addressing MLOps projects. The application of DevOps principles to ML and the use of MLOps in the industrial environment are still little discussed topics at the academic level. Current literature is mostly disconnected and sporadic. This paper is intended as a literature review to systematize and add clarity to the definition and methods of MLOps. The paper aims to define a high-level strategy for dealing with MLOps projects; the goal of future work is to apply our proposed methodology to use cases such as biomedical imaging and finance. Experimental work will be required to test the pipeline defined in this manuscript.

Traditionally, data preparation, model training and testing, and performance comparison are key points of traditional pipelines. In this work, we have stressed the importance of many other, no less important aspects, such as continuous monitoring, sustainability issues, etc. Following well-defined guidelines is the only way to allow the traceability and reproducibility of the results obtained in an Open Science context. For this reason, it is crucial to use systematic procedures for greater cohesion in the scientific community to follow clear and clean pipelines in MLOps. The remaining challenge for the community is to try to apply an ML methodology to an end-to-end use case trying to go through each point of this methodology and show what happens if some phases are not used. Specific areas, such as biomedicine, finance, cyber-security, manufacturing [97], can greatly benefit from adopting MLOps, and we believe the pipeline defined in this paper can bring advantages over traditional practices.

According to Fortune Business Insights, the global Machine Learning market is expected to grow from \$15.50 billion in 2021 to \$152.24 billion in 2028 with a compound annual growth rate of 38.6% over the forecast period. MLOps aims to create long-term ML solutions, reducing maintenance costs, and monitoring and optimizing workflows. Understanding and intercepting new challenges and trends such as the emerging MLOps will provide a strong competitive advantage to companies adopting this solution [98]

ABBREVIATION TERMS

ML	Machine Learning.
MLOps	Machine Learning Operations.
AI	Artificial Intelligence.
XAI	eXplainable AI.
STEM	Science, Technology, Engineering and Mathematics.
DevOps	Development Operations.
DL	Deep Learning.
ROI	Return on Investments.
CI	Continuos Integration.
CD	Continuos Delivery.
CT	Continuos Training.
CRISP-DM	CRoss-Industry Standard Process for Data Mining.
KPI	Key Performance Indicator.
MSE	Mean Squared Error.
ROS	Return on Sales.
REST	REpresentational State Transfer.
OKR	Objective and Key Result.
API	Application Programming Interface.
NLP	Natural Language Processing.

REFERENCES

[1] R. Akita, A. Yoshihara, T. Matsubara, and K. Uehara, "Deep learning for stock prediction using numerical and textual information," in *Proc. IEEE/ACIS 15th Int. Conf. Comput. Inf. Sci. (ICIS)*, Jun. 2016, pp. 1–6.

[2] M. C. Fiorentino, E. Cipolletta, E. Filippucci, W. Grassi, E. Frontoni, and S. Moccia, "A deep-learning framework for metacarpal-head cartilage-thickness estimation in ultrasound rheumatological images," *Comput. Biol. Med.*, vol. 141, Feb. 2022, Art. no. 105117.

[3] M. Jamshidi, A. Lalbakhsh, J. Talla, Z. Peroutka, F. Hadjilooei, P. Lalbakhsh, M. Jamshidi, L. La Spada, M. Mirmozafari, M. Dehghani, A. Sabet, S. Roshani, S. Roshani, N. Bayat-Makou, B. Mohamadzade, Z. Malek, A. Jamshidi, S. Kiani, H. Hashemi-Dezaki, and W. Mohyuddin, "Artificial intelligence and COVID-19: Deep learning approaches for diagnosis and treatment," *IEEE Access*, vol. 8, pp. 109581–109595, 2020.

[4] M. B. Jamshidi, A. Lalbakhsh, J. Talla, Z. Peroutka, S. Roshani, V. Matousek, S. Roshani, M. Mirmozafari, Z. Malek, L. L. Spada, and A. Sabet, "Deep learning techniques and COVID-19 drug discovery: Fundamentals, state-of-the-art and future directions," in *Emerging Technologies During the Era of COVID-19 Pandemic*. Cham, Switzerland: Springer, 2021, pp. 9–31.

[5] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *J. Field Robot.*, vol. 37, no. 3, pp. 362–386, Apr. 2020.

[6] A. Roy, J. Sun, R. Mahoney, L. Alonzi, S. Adams, and P. Beling, "Deep learning detecting fraud in credit card transactions," in *Proc. Syst. Inf. Eng. Design Symp. (SIEDS)*, Apr. 2018, pp. 129–134.

[7] J. Frizzo-Barker, P. A. Chow-White, M. Mozafari, and D. Ha, "An empirical study of the rise of big data in business scholarship," *Int. J. Inf. Manage.*, vol. 36, no. 3, pp. 403–413, Jun. 2016.

[8] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Inf. Softw. Technol.*, vol. 50, nos. 9–10, pp. 833–859, 2008.

[9] T. J. Gandomani, Z. Tavakoli, H. Zulzalil, and H. K. Farsani, "The role of project manager in agile software teams: A systematic literature review," *IEEE Access*, vol. 8, pp. 117109–117121, 2020.

[10] Z. Luqing, "Research on the impact of standardization on economic growth," *Management*, vol. 9, no. 6, pp. 236–241, 2021.

[11] K. Blind, A. Jungmittag, and A. Mangelsdorf, "The economic benefits of standardization," DIN German Inst. Standardization, Berlin, Germany, Tech. Rep., 2011.

[12] O. Khalaj, M. B. Jamshidi, E. Saebnoori, B. Mašek, C. Štadler, and J. Svoboda, "Hybrid machine learning techniques and computational mechanics: Estimating the dynamic behavior of oxide precipitation hardened steel," *IEEE Access*, vol. 9, pp. 156930–156946, 2021.

[13] S. Alla and S. K. Adari, "What is MLOps?" in *Beginning MLOps With MLFlow*. Cham, Switzerland: Springer, 2021, pp. 79–124.

[14] S. Mäkinen, H. Skogström, E. Laaksonen, and T. Mikkonen, "Who needs MLOps: What data scientists seek to accomplish and how can MLOps help?" 2021, *arXiv:2103.08942*.

[15] U. Bhatt, M. Andrus, A. Weller, and A. Xiang, "Machine learning explainability for external stakeholders," 2020, *arXiv:2007.05408*.

[16] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, "DevOps," *IEEE Softw.*, vol. 33, no. 3, pp. 94–100, Apr. 2016.

[17] S. M. Mohammad, "DevOps automation and agile methodology," in *Proc. Int. J. Creative Res. Thoughts (IJCRT)*, 2017, pp. 2320–2882.

[18] M. Borg, R. Jabangwe, S. Åberg, A. Ekblom, L. Hedlund, and A. Lidfeldt, "Test automation with grad-CAM Heatmaps—A future pipe segment in MLOps for vision AI?" in *Proc. IEEE Int. Conf. Softw. Test., Verification Validation Workshops (ICSTW)*, Apr. 2021, pp. 175–181.

[19] G. Fursin, "The collective knowledge project: Making ML models more portable and reproducible with open APIs, reusable best practices and MLOps," 2020, *arXiv:2006.07161*.

[20] J. P. Gujjar and V. N. Kumar, "Demystifying mlops for continuous delivery of the product," *Asian J. Adv. Res.*, vol. 18, pp. 19–23, Feb. 2022.

[21] S. Moreschini, F. Lomio, D. Hästbacka, and D. Taibi, "MLOps for evolvable AI intensive software systems," Tech. Rep.

[22] M. Vizard, Kirsch, and, "DevOps," Tech. Rep., Nov. 2021.

[23] I. Karamitsos, S. Albarhami, and C. Apostolopoulos, "Applying DevOps practices of continuous automation for machine learning," *Information*, vol. 11, no. 7, p. 363, Jul. 2020.

[24] B. S. Farroha and D. L. Farroha, "A framework for managing mission needs, compliance, and trust in the DevOps environment," in *Proc. IEEE Mil. Commun. Conf.*, Oct. 2014, pp. 288–293.

[25] J.-P. Correa-Baena, K. Hippalgaonkar, J. van Duren, S. Jaffer, V. R. Chandrasekhar, V. Stevanovic, C. Wadia, S. Guha, and T. Buonassisi, "Accelerating materials development via automation, machine learning, and high-performance computing," *Joule*, vol. 2, no. 8, pp. 1410–1420, Aug. 2018.

[26] J. Mizgajski, A. Szymczak, M. Morzy, Ł. Augustyniak, P. Szymański, and P. Żelasko, "Return on investment in machine learning: Crossing the chasm between academia and business," *Found. Comput. Decis. Sci.*, vol. 45, no. 4, pp. 281–304, Dec. 2020.

[27] W. EckersonGroup, "Eckerson," Tech. Rep., 2022.

[28] Y. Zhao, "Machine learning in production: A literature," Tech. Rep., 2021.

[29] E. D. S. Nascimento, I. Ahmed, E. Oliveira, M. P. Palheta, I. Steinmacher, and T. Conte, "Understanding development process of machine learning systems: Challenges and solutions," in *Proc. ACM/IEEE Int. Symp. Empirical Softw. Eng. Meas. (ESEM)*, Sep. 2019, pp. 1–6.

[30] J. Dalzochio, R. Kunst, E. Pignaton, A. Binotto, S. Sanyal, J. Favilla, and J. Barbosa, "Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges," *Comput. Ind.*, vol. 123, Dec. 2020, Art. no. 103298.

[31] L. Scotton, "Engineering framework for scalable machine learning operations," Tech. Rep., 2021.

[32] E. Breck, S. Cai, E. Nielsen, M. Salib, and D. Sculley, "The ML test score: A rubric for ML production readiness and technical debt reduction," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 1123–1132.

[33] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 2503–2511.

[34] H. Muccini and K. Vaidhyanathan, "Software architecture for ML-based systems: What exists and what lies ahead," 2021, *arXiv:2103.07950*.

[35] J. Klaise, A. Van Looveren, C. Cox, G. Vacanti, and A. Coca, "Monitoring and explainability of models in production," 2020, *arXiv:2007.06299*.

[36] M. Arnold, J. Boston, M. Desmond, E. Duesterwald, B. Elder, A. Murthi, J. Navratil, and D. Reimer, "Towards automating the AI operations lifecycle," 2020, *arXiv:2003.12808*.

[37] T. Granlund, A. Kopponen, V. Stirbu, L. Myllyaho, and T. Mikkonen, "MLOps challenges in multi-organization setup: Experiences from two real-world cases," 2021, *arXiv:2103.08937*.

[38] T. Granlund, V. Stirbu, and T. Mikkonen, "Towards regulatory-compliant MLOps: Oravizio's journey from a machine learning experiment to a deployed certified medical product," *Social Netw. Comput. Sci.*, vol. 2, no. 5, pp. 1–14, Sep. 2021.

[39] O. Spjuth, J. Frid, and A. Hellander, "The machine learning life cycle and the cloud: Implications for drug discovery," *Expert Opinion Drug Discovery*, vol. 16, no. 9, pp. 1–9, 2021.

[40] M. Zaharia, A. Chen, A. Davidson, A. Ghodsi, S. A. Hong, A. Konwinski, S. Murching, T. Nykodym, P. Ogilvie, and M. Parkhe, "Accelerating the machine learning lifecycle with MLflow," *IEEE Data Eng. Bull.*, vol. 41, no. 4, pp. 39–45, Jun. 2018.

[41] K. Org, "Kedro-org/Kedro: A Python framework for creating reproducible, maintainable and modular data science code," Tech. Rep.

[42] D. Hudgeon and R. Nichol, "Machine learning for business: Using Amazon SageMaker and Jupyter," Tech. Rep., 2020.

[43] V. J. Reddi, C. Cheng, D. Kanter, P. Mattson, G. Schmuelling, C.-J. Wu, B. Anderson, M. Breughe, M. Charlebois, and W. Chou, "MLPerf inference benchmark," in *Proc. ACM/IEEE 47th Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2020, pp. 446–459.

[44] C. Li, A. Dakkak, J. Xiong, W. Wei, L. Xu, and W.-M. Hwu, "XSP: Across-stack profiling and analysis of machine learning models on GPUs," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS)*, May 2020, pp. 326–327.

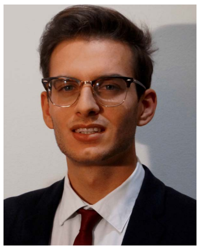
[45] T. Ben-Nun, M. Besta, S. Huber, A. N. Ziogas, D. Peter, and T. Hoefler, "A modular benchmarking infrastructure for high-performance and reproducible deep learning," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS)*, May 2019, pp. 66–77.

[46] C. Shearer, "The CRISP-DM model: The new blueprint for data mining," *J. Data Warehousing*, vol. 5, no. 4, pp. 13–22, 2000.

[47] H. Ahmed, D. Tahseen, W. Haider, M. Asad, S. Nand, and S. Kamran, "Establishing standard rules for choosing best KPIs for an E-commerce business based on Google analytics and machine learning technique," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 5, pp. 12–24, 2017.

[48] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," 2019, *arXiv:1903.12261*.
[49] W. Samek, T. Wiegand, and K.-R. Müller, "Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models," 2017, *arXiv:1708.08296*.
[50] Z. Wang, Y. Lai, Z. Liu, and J. Liu, "Explaining the attributes of a deep learning based intrusion detection system for industrial control networks," *Sensors*, vol. 20, no. 14, p. 3817, Jul. 2020.
[51] S. Rabiul Islam, W. Eberle, S. Khaled Ghafoor, and M. Ahmed, "Explainable artificial intelligence approaches: A survey," 2021, *arXiv:2101.09429*.
[52] B. Fitzgerald and K.-J. Stol, "Continuous software engineering and beyond: Trends and challenges," in *Proc. 1st Int. Workshop Rapid Continuous Softw. Eng. (RCoSE)*, 2014, pp. 1–9.
[53] G. Symeonidis, E. Nerantzis, A. Kazakis, and G. A. Papakostas, "MLOps–definitions, tools and challenges," 2022, *arXiv:2201.00162*.
[54] N. Hewage and D. Meedeniya, "Machine learning operations: A survey on MLOps tool support," 2022, *arXiv:2202.10169*.
[55] *AI Usecases & Tools to Grow Your Business*, Jan. 2022.
[56] M. J. Willemink, W. A. Koszek, C. Hardell, J. Wu, D. Fleischmann, H. Harvey, L. R. Folio, R. M. Summers, D. L. Rubin, and M. P. Lungren, "Preparing medical imaging data for machine learning," *Radiology*, vol. 295, no. 1, pp. 4–15, Apr. 2020.
[57] T. Kulesza, S. Amershi, R. Caruana, D. Fisher, and D. Charles, "Structured labeling for facilitating concept evolution in machine learning," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, Apr. 2014, pp. 3075–3084.
[58] T. van der Weide, D. Papadopoulos, O. Smirnov, M. Zielinski, and T. van Kasteren, "Versioning for end-to-end machine learning pipelines," in *Proc. 1st Workshop Data Manage. End-End Mach. Learn.*, May 2017, pp. 1–9.
[59] A. Zheng and A. Casari, *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. Newton, MA, USA: O'Reilly Media, 2018.
[60] A. W. Long, J. Zhang, S. Granick, and A. L. Ferguson, "Machine learning assembly landscapes from particle tracking data," *Soft Matter*, vol. 11, no. 41, pp. 8141–8153, 2015.
[61] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 2623–2631.
[62] D. Golovin, B. Solnik, S. Moitra, G. Kochanski, J. Karro, and D. Sculley, "Google vizier: A service for black-box optimization," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 1487–1495.
[63] L. Baier, F. Jöhren, and S. Seebacher, "Challenges in the deployment and operation of machine learning in practice," in *Proc. ECIS*, 2019, pp. 1–15.
[64] O. Boursalie, R. Samavi, and T. E. Doyle, "M4CVD: Mobile machine learning model for monitoring cardiovascular disease," *Proc. Comput. Sci.*, vol. 63, no. 2, pp. 384–391, 2015.
[65] M. Syafrudin, G. Alfian, N. Fitriyani, and J. Rhee, "Performance analysis of IoT-based sensor, big data processing, and machine learning model for real-time monitoring system in automotive manufacturing," *Sensors*, vol. 18, no. 9, p. 2946, Sep. 2018.
[66] R. Ashmore, R. Calinescu, and C. Paterson, "Assuring the machine learning lifecycle: Desiderata, methods, and challenges," *ACM Comput. Surveys*, vol. 54, no. 5, pp. 1–39, Jun. 2022.
[67] A. F. V. Maya, "The state of MLOps," Tech. Rep., 2021.
[68] M. Badawy, A. A. A. El-Aziz, A. M. Idress, H. Hefny, and S. Hossam, "A survey on exploring key performance indicators," *Future Comput. Informat. J.*, vol. 1, nos. 1–2, pp. 47–52, Dec. 2016.
[69] S. Agrawal and A. Mittal, "MLOps: 5 steps to operationalize machine-learning models—AI4," Tech. Rep.
[70] Y. Li, X. Yu, and N. Koudas, "Data acquisition for improving machine learning models," 2021, *arXiv:2105.14107*.
[71] P. Ruf, M. Madan, C. Reich, and D. Ould-Abdeslam, "Demystifying MLOps and presenting a recipe for the selection of open-source tools," *Appl. Sci.*, vol. 11, no. 19, p. 8861, Sep. 2021.
[72] M. Treveil, *Introducing MLOps: How to Scale Machine Learning in the Enterprise*. Newton, MA, USA: O'Reilly, 2020.
[73] J. Bosch, "Continuous software engineering: An introduction," in *Continuous Software Engineering*. Cham, Switzerland: Springer, 2014, pp. 3–13.
[74] M. Leppänen, S. Mäkinen, M. Pagels, V. P. Eloranta, J. Itkonen, M. V. Mäntylä, and T. Männistö, "The highways and country roads to continuous deployment," *IEEE Softw.*, vol. 32, no. 2, pp. 64–72, Mar. 2015.
[75] L. E. Lwakatare, I. Crnkovic, E. Rånge, and J. Bosch, "From a data science driven process to a continuous delivery process for machine learning systems," in *Proc. Int. Conf. Product-Focused Softw. Process Improvement*. Cham, Switzerland: Springer, 2020, pp. 185–201.
[76] J. F. Smart, *Jenkins: The Definitive Guide: Continuous Integration for Masses*. Newton, MA, USA: O'Reilly Media, 2011.
[77] S. Chacon and B. Straub, *Pro Git*. New York, NY, USA: Apress, 2014.
[78] D. Merkel, "Docker: Lightweight Linux containers for consistent development and deployment," *Linux J.*, vol. 2014, no. 239, p. 2, 2014.
[79] *PM. Helm*, 2018.
[80] (2017). *Kubernetes Manual*. Accessed: Dec. 4, 2017. [Online]. Available: https://kubernetes.io/
[81] I. Weber, S. Nepal, and L. Zhu, "Developing dependable and secure cloud applications," *IEEE Internet Comput.*, vol. 20, no. 3, pp. 74–79, May 2016.
[82] P. Webteam, "Resources: Puppet," Tech. Rep.
[83] L. Chen, "Continuous delivery: Huge benefits, but challenges too," *IEEE Softw.*, vol. 32, no. 2, pp. 50–54, Mar. 2015.
[84] B. Liu, "Lifelong machine learning: A paradigmphfor continuous learning," *Frontiers Comput. Sci.*, vol. 11, no. 3, pp. 359–361, 2017.
[85] A. Komolafe, "Retraining model during deployment: Continuous training and continuous testing," Tech. Rep., Dec. 2021.
[86] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern Recognit.*, vol. 45, no. 1, pp. 521–530, 2012.
[87] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, Nov. 2015.
[88] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artif. Intell.*, vol. 267, pp. 1–38, Feb. 2019.
[89] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.
[90] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, "Explainable AI: A review of machine learning interpretability methods," *Entropy*, vol. 23, no. 1, p. 18, Dec. 2020.
[91] L. F. W. Anthony, B. Kanding, and R. Selvan, "Carbontracker: Tracking and predicting the carbon footprint of training deep learning models," 2020, *arXiv:2007.03051*.
[92] M. Płoszaj-Mazurek, E. Ryńska, and M. Grochulska-Salak, "Methods to optimize carbon footprint of buildings in regenerative architectural design with the use of machine learning, convolutional neural network, and parametric design," *Energies*, vol. 13, no. 20, p. 5289, Oct. 2020.
[93] T. Wiedmann and J. Minx, "A definition of 'carbon footprint,'" *Ecol. Econ. Res. trends*, vol. 1, pp. 1–11, Mar. 2008.
[94] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in NLP," 2019, *arXiv:1906.02243*.
[95] G. Tamburrini, "The AI carbon footprint and responsibilities of AI scientists," *Philosophies*, vol. 7, no. 1, p. 4, Jan. 2022.
[96] U. European Commission, "White paper on artificial intelligence—European commission," Tech. Rep.
[97] A. Shafiei, M. Jamshidi, F. Khani, J. Talla, Z. Peroutka, R. Gantassi, M. Baz, O. Cheikhrouhou, and H. Hamam, "A hybrid technique based on a genetic algorithm for fuzzy multiobjective problems in 5G, Internet of Things, and mobile edge computing," *Math. Problems Eng.*, vol. 2021, pp. 1–14, Oct. 2021.
[98] M. B. Jamshidi, N. Alibeigi, N. Rabbani, B. Oryani, and A. Lalbakhsh, "Artificial neural networks: A powerful tool for cognitive science," in *Proc. IEEE 9th Annu. Inf. Technol., Electron. Mobile Commun. Conf. (IEMCON)*, Nov. 2018, pp. 674–679.

MATTEO TESTI is currently pursuing the Ph.D. degree in computer engineering with the Medical Statistic and Molecular Epidemiology Unit, University of Biomedical Campus, Rome, Italy. He is also an Entrepreneur with a strong background in data science with a focus on deep learning. He founded DeepLearningItalia the biggest e-learning platform in the artificial intelligence area in the Italian language. He was one of the technical writers for the Artificial Intelligence Italian white paper. Since 2019, he has been an Adjunct Professor with the University of Rome Tor Vergata, Rome.



MATTEO BALLABIO was born in Carate Brianza, Monza and Brianza, in January 1999. He received the B.Sc. degree in biomedical engineering from Bergamo University, Italy, in December 2021, where he is currently pursuing the M.Sc. degree in management engineering. He has been a Collaborator at DeepLearningItalia, since September 2021. He is the Contributor of the dataset "DBB Distorted Brain Benchmark" (BrainLife.io) used in the publication "Automatic Tissue Segmentation with Deep Learning in Patients with Congenital or Acquired Distortion of Brain Anatomy" published on Springer. He received the B.Sc. thesis entitled: "Creation of an open dataset for the evaluation of the segmentation of MRI images in the case of patients with severe distortions of brain anatomy" in collaboration with the Research Center Fondazione Bruno Kessler (FBK), Trento (IT).

EMANUELE FRONTONI (Member, IEEE) is currently a Full Professor in computer science with the University of Macerata and the Co-Director of the VRAI Laboratory, Marche Polytechnic University. His research interests include computer vision and artificial intelligence with applications in robotics, video analysis, human behavior analysis, and digital humanities. He is involved in several industrial research and development projects in collaboration with ICT and mechatronics companies in the field of artificial intelligence. He is a member of the European Association for Artificial Intelligence, the European AI Alliance, and the International Association for Pattern Recognition.

GIULIO IANNELLO (Member, IEEE) received the Graduate degree in electronic engineering from the Politecnico di Milano, in 1981, and the Ph.D. degree in computer science and computer engineering from the University of Napoli Federico II, in 1987. Currently, he is a Full Professor in computer science and computer engineering with the University Campus Bio-Medico di Roma, where is responsible of the Research Group on Computer Systems and Bionformatics. His current research interests include data, signal and image processing for biomedical and biological applications, high performance computing, design, and analysis of parallel algorithms. He has published over 150 journals and conference papers in these and related areas. He is a member of several program committees of international conferences and referee of international journals. He is a member of the ACM, CVPL, and SIBIM.

SARA MOCCIA was born in Bari, in September 1990. She received the B.Sc. degree, in 2012, the M.Sc. degree *(cum laude)* in biomedical engineering from the Politecnico di Milano, Milan, Italy, in December 2014, and the joint Ph.D. (European) degree *(cum laude)* in bioengineering from the Istituto Italiano di Tecnologia, Genoa, Italy, in May 2018, and the Politecnico di Milano, for which she was awarded by the Italian Group of Bioengineering (Gruppo Nazionale di Bioingegneria). During her Ph.D. degree, she was hosted with the Computer-Assisted Medical Interventions Laboratory, German Cancer Research Center, Heidelberg, Germany. From May 2018 to January 2021, she was a Postdoctoral Researcher with Università Politecnica delle Marche, Ancona, Italy, and an Affiliated Researcher with the Istituto Italiano di Tecnologia. In 2021, she was also a Visiting Researcher with the University of Minho, Braga, Portugal. She is currently an Assistant Professor at Scuola Superiore Sant'Anna, Pisa, Italy, and an Adjunct Professor with Università Politecnica delle Marche.

PAOLO SODA (Member, IEEE) is currently a Full Professor in computer science and computer engineering with the University Campus Bio-Medico di Roma, and he is also a Visiting Professor in biomedical engineering and AI with the Department of Radiation Sciences, Umeå University, Sweden. He is also a Vice-Coordinator of the Health and Life Sciences specialization area of the National Ph.D. program in AI. His research interests include AI, machine learning, and big data analytics, with applications to data, signals, 2D and 3D image, and video processing and analysis. He was the Team Leader of the Research Groups that won the two international competitions: "COVID CXR Hackathon" (2022 Dubai Expo) and "All against COVID-19: Screening X-ray Images for COVID-19 Infection" (IEEE 2021). He is a member of CVPL and SIBIM, and the chairs of the IEEE International Technical Committee for Computational Life Sciences.

GENNARO VESSIO (Member, IEEE) received the M.Sc. degree (Hons.) in computer science, in 2013, and the Ph.D. degree in computer science and mathematics from the Department of Computer Science, University of Bari, Italy, in 2017. He is currently an Assistant Professor with the University of Bari. His current research interests include pattern recognition, machine and deep learning, computer vision, and their application to several domains, including e-health, drone vision, and digital humanities. He is a member of the Editorial Board of *International Journal of Intelligent Systems* and *Computational Intelligence and Neuroscience*. He is involved in the organization of scientific events, the most recent of which was the Second International Workshop on Fine Art Pattern Extraction and Recognition within ICIAP. He regularly serves as a reviewer for many international journals published by high-level publishers, including Elsevier, IEEE, and Springer, and as a member of the program committee of many international conferences. He is regularly involved in the teaching activities of his department and has supervised dozens of graduate students in computer science. He is currently a member of the IEEE Computer Society, the IEEE Computational Intelligence Society, the INdAM-GNCS Society, the IAPR Technical Committee 19 (Computer Vision for Cultural Heritage Applications), the AIxIA association, the CINI-AIIS Laboratory, the CITEL Telemedicine Research Center, the CEDITH Digital Heritage Research Center, GRIN, MIR Laboratories, and the REPRISE Register of Experts.