A Preliminary Study of MLOps Practices in GitHub

Fabio Calefato
University of Bari
Bari, Italy
fabio.calefato@uniba.it

Filippo Lanubile
University of Bari
Bari, Italy
filippo.lanubile@uniba.it

Luigi Quaranta
University of Bari
Bari, Italy
luigi.quaranta@uniba.it

Abstract

The rapid and growing popularity of machine learning (ML) applications has led to an increasing interest in MLOps, i.e., the practice of continuous integration and deployment (CI/CD) of ML-enabled systems. Since changes may affect not only the code but also the ML model parameters and the data themselves, the automation of traditional CI/CD needs to be extended to manage model retraining in production. Here we present an initial investigation of the MLOps practices implemented in a set of ML-enabled systems retrieved from GitHub. Our preliminary results suggest that the current adoption of MLOps workflows in open-source GitHub projects is rather limited. Issues are also identified, which can guide future research work.

1 Introduction

ML-enabled systems, i.e., software systems incorporating machine learning (ML) models, are receiving more and more attention from both researchers and practitioners (Ozkaya, 2020; Lewis et al., 2021; Nahar et al., 2021). Accordingly, there is also an increasing interest in MLOps (Makinen et al., 2021; Treveil et al., 2020), i.e., the development of solutions for the rapid delivery and deployment of ML models. Because ML models are the core part of larger software systems, MLOps builds on DevOps and GitOps practices (Ebert et al., 2016; Limoncelli, 2018), and introduces additional actions that are specific to machine learning.

Despite the increasing popularity of MLOps, there is a lack of studies on its adoption and impact on ML-enabled systems. To fill this gap, we start from open source projects available on GitHub, asking the following research questions:

RQ1. *How common is workflow automation in ML-enabled systems hosted on GitHub?*

RQ2. *What type of events are used to trigger MLOps workflows?*

RQ3. *What are the most frequently enacted tasks?*

We answer these questions by means of an explorative archival study that mixes quantitative and qualitative analysis, focusing on GitHub Actions and CML, two freely available solutions. GitHub Actions is a popular continuous integration and continuous delivery (CI/CD) platform backed by GitHub, which automates development workflows through an event-driven API that. Workflows are defined by storing YAML files (e.g., `build.yml`) checked-in to a project's `.github/workflows/` directory, whose *actions* will typically run when triggered by an *event* occurring in the repository, such as a developer creating a pull request or pushing a commit. CML is an open-source, command-line interface tool for implementing CI/CD in machine learning projects hosted on GitHub or GitLab. In

NeurIPS 2022 Workshop on Challenges in Deploying and Monitoring Machine Learning Systems.

GitHub, CML builds upon the GITHUB ACTIONS infrastructure, with developers defining *commands* within a `cml.y*ml` file to be stored in the same `.github/workflows/` directory.

In the rest of the paper, we use the term *task* as an abstraction of *action* in GITHUB ACTIONS and *command* in CML.

2 Dataset Construction

GITHUB ACTIONS workflows. We decided to focus on GITHUB ACTIONS as it is the default workflow automation tool offered by GitHub to open-source projects. To assemble the experimental dataset, we started from the one used in recent work on the use of GITHUB ACTIONS (Kinsman et al., 2021), comprising 446,862 GitHub repositories. To keep low-quality projects out of our dataset, we decided to analyze only repositories having more than one star (i.e., 183,127 out of the 446,862 repositories from the original set), which had been active for at least six months after the public release of GITHUB ACTIONS (November 2019). First, we filtered out all repositories whose last-commit date preceded May 2020. In addition, to restrict our selection to ML-related repositories, we adopted a strategy similar to the one used in Biswas et al. (2019) and leveraged the GitHub API to retrieve the descriptions and topics of the repositories resulting from the filtering above. Then, we sought ML systems-related keywords contained therein (e.g., *'machine learning'*, *'deep learning'*, *'neural network'*, *'image processing'*, etc.). After applying the keyword-based filtering, we obtained a set of 2,516 repositories. Afterwards, we checked the adoption of GITHUB ACTIONS by verifying the existence of YAML files in the `./github/workflows` directory. Only 155 contain at least one GITHUB ACTIONS workflow. Contextually, we downloaded all the available 399 workflows, 2 of which resulted invalid. In conclusion, the first experimental dataset — hereafter referred to as the GITHUB ACTIONS dataset — consists of 397 valid workflows extracted from 155 repositories.

CML workflows. To build the second dataset of projects containing CML workflows, we used the advanced search web interface to search globally across all of GitHub for repositories containing *.github/workflows/cml.y*ml* files. We decided to focus on CML because it is designed as a natural, ML-oriented extension of GITHUB ACTIONS and it is also tightly integrated with DVC, one of the most popular data version control tools available to date (Barrak et al., 2021). The query returned a list of only 36 candidates. Given the few hits, in this case, we did not apply any quality filtering (e.g., number of stars). Still, we manually vetted the list to exclude non-relevant results. In particular, six repositories were discarded because they contain no ML models (2), no code (3), and an empty workflow file (1); we also discarded one repository that is merely the implementation of one of the tutorials available on the CML website. We then downloaded the workflows from the retained repositories. In conclusion, our second dataset — hereafter referred to as the CML dataset — contains 41 CML workflows extracted from 29 repositories, of which 38 are valid.

3 Data Analysis Method

To answer RQ1, we characterized the general adoption of GITHUB ACTIONS and CML in our datasets by manually verifying that the selected repositories actually qualify as ML-enabled systems.

For RQ2, we computed the frequency of GitHub events used to trigger the workflows in our dataset.

Regarding RQ3, to infer the most frequent tasks, we sought patterns among recurring actions and shell commands from the collected workflows. Specifically, with the term *'task'* here we refer to any logical unit of work that is typically found within an ML development workflow (e.g., 'launching a model-retraining job' or 'deploying an ML-enabled component to production,'). Therefore, to uncover patterns that may possibly constitute high-level tasks, we started by analyzing the most frequently used actions. Accordingly, we first computed the descriptive statistics of the actions found in our datasets, while also identifying the occurrence of those containing the words `'cml'` or 'docker,' which we consider as traces of typical MLOps practices adoption (i.e., experiment tracking and ML models deployment, respectively). Then, we leveraged the *Apriori* algorithm to compute the set of frequently co-occurring actions, thus generating a collection of transactions.A transaction corresponds to the collection of actions observed together in the context of a single workflow.Finally, we performed a qualitative analysis of the workflows by manually inspecting the most interesting ones, i.e., those belonging to repositories that contain ML-enabled software components.

4 Results

4.1 RQ1. How common is workflow automation in ML-enabled systems hosted on GitHub?

GITHUB ACTIONS. Our GITHUB ACTIONS dataset comprises 397 valid GITHUB ACTIONS workflows distributed across 155 repositories. The median number of workflows contained in each repository is 2, while the third quartile of the distribution is 3 and the maximum number of workflows found in a repository is 14. We manually verified the results of the filtering process with a manual inspection by which we checked that each repository actually contained an ML-enabled system or an ML-enabled software component to be integrated in a larger system. Most of the repositories (105) have been misclassified as ML-related projects; the remaining repositories (50) are ML libraries or frameworks (e.g., `dmlc/xgboost`, `h2oai/h2o-dev`). We decided to discard these kinds of repositories as – apart from their ML-oriented goal – they actually look no different from traditional software projects, with which they share the typical engineering requirements and needs (Ozkaya, 2020). By looking at the misclassified instances, we noticed that the '*machine learning*' and '*artificial intelligence*' topic labels are often used as buzzwords due to the current hype. Only one repository can be considered an actual example of ML-enabled system (e.g., `tesseract-ocr/tesseract`). However, the related GITHUB ACTIONS workflows are devoted to testing and benchmarking the application as a whole and none of them addresses ML tasks directly.

Having not found relevant workflows in the GITHUB ACTIONS dataset, we decided to discard it. As such, the analytical results hereafter presented are exclusively referred to the CML dataset discussed next. We speculate that most of the professional collaboration around ML-enabled components and systems might currently be happening within companies and private research institutions given that ML models constitute a valuable economical asset. Consequently, we argue that ML-enabled systems may be more frequently found in private GitHub repositories.

CML. The CML dataset comprises 38 valid GITHUB ACTIONS workflows distributed across 29 repositories. The median number of workflows contained in each repository is 1 and the third quartile of the distribution is still 1; the maximum number of workflows found in a repository is 4. Also in this case, we manually vetted the dataset. We classified most of the repositories from this collection (24) as *test-driving* (i.e., repositories containing proof-of-concept ML projects, likely used by GitHub users to test-drive the CML library). A couple of repositories are explicitly linked to educational material, i.e., a university ML course (`tue-5ARA0/mlops-demo-live`) and a technical blog post (`amitvkulkarni/Bring-DevOps-to-Machine-Learning-with-CML`); as such, we classify them as *educational*. Finally, in 3 repositories we found small-size ML projects (i.e., up to three contributors), to which we refer as *ML component*: the first (`AscendNTNU/perception_testing_21`) is a workspace for building models for a robot operating system; the second (`cheesama/morphine`) an entity classifier; the third (`mozartofmath/AmharicSpeechToText`) a speech-to-text engine.

We observe again a substantial lack of production-grade ML projects containing traces of MLOps practices. However, all repositories from this dataset contain CML workflows, each of which is specifically defined to accomplish one or more MLOps tasks. As such, despite many of them have testing or educational purposes, all the repositories from this dataset were deemed relevant.

4.2 RQ2. What type of events are used to trigger MLOps workflows?

Most of the workflows from our dataset are triggered by `git push` (79%) and `pull_request` events (18%). Moreover, these events co-occur as triggers in 13% of the workflows. Instead, GitHub-managed events are rarely used to activate workflows. Only 3 of them start as a consequence of a new `issue_comment` and just 2 are triggered by GitHub's `release` event. The `schedule` event – allowing workflows activation at specific times – is also used only twice. While it is not surprising that the native `git` events `push` and `pull_request` are the most commonly used triggers for projects hosted on GitHub, we expected a prevalence of workflows activated by `pull_request` and `release` events. However, typical MLOps tasks are computationally heavy and time consuming (e.g., model re-training) and, as such, presumably expensive to run every time a new push operation is performed.

4.3 RQ3. What are the most frequently enacted tasks?

To identify potential ML-related tasks accomplished within the collected workflows, we analyzed the pre-defined actions used in each of them. The total number of distinct actions retrieved in the

Table 1: Frequently co-occurring actions with support > 0.07 (i.e., co-occurring in 3+ workflows).

Action 1	Action 2	Action 3	Support
actions/checkout	iterative/setup-cml		0.47 (18 workflows)
actions/checkout	actions/setup-python		0.37 (14 workflows)
actions/setup-cml	actions/setup-python		0.34 (13 workflows)
actions/checkout	actions/setup-python	actions/setup-cml	0.34 (13 workflows)
actions/checkout	aws-actions/configure-aws-credentials		0.11 (4 workflows)
actions/checkout	actions/setup-go		0.08 (3 workflows)

CML dataset is 15. By using the *Apriori* algorithm, we computed the frequent sets of actions with support higher than 0.07 (i.e., appearing in at least 3 workflows); the results are reported in Table 1. The top-4 frequent itemsets combine `actions/checkout` with `actions/setup-cml` and `actions/setup-python`. If we limit our analysis to the workflows containing '*cml*' in their filename, we find that only 6 distinct actions are used. Table 2 presents each of them together with their frequencies. A brief definition is available in Appendix A. Every CML workflow uses `actions/checkout` to make the contents of the repository available in the actions runner; then, about half of them set up CML and Python using pre-defined actions. All these actions are primarily used to set up the CI environment and accomplish preliminary configuration steps.

Table 2: Actions found within CML workflows.

Action	Frequency
actions/checkout	28
iterative/setup-cml	17
actions/setup-python	13
iterative/setup-dvc	2
ros-industrial/industrial_ci	1
aws-actions/configure-aws-credentials	1

We now delved into the analyses of shell commands extracted from `run` attributes. Of the 38 GITHUB ACTIONS workflows from the CML dataset, 28 present shell commands containing '*cml*' as a sub-string. CML offers a few commands to support the implementation of CI/CD in ML projects. A brief definition is available in Appendix B. All 28 workflows containing a CML command use `send-comment` to decorate a commit or pull-request message with model training and evaluation metrics. In 17 cases (~63%), the `publish` command is used along with `send-comment` to add an image to the Markdown-formatted message. The adoption of `runner` in (`ibrahimkaratas88/cml_cloud_try`) is worth mentioning: we found that it is used to set up an AWS virtual machine, transfer the data on the cloud machine via DVC, and report the related results via the `send-comment` command. Overall, none of these workflows involve model training or re-training tasks. Thus, to identify end-to-end MLOps pipelines, we also sought the use of Docker commands. Within our dataset, Docker is used just once in a repository classified as a *test-driving*. This workflow is executed only when a message containing the string *'/dockerize'* is published as a comment in an issue or pull request. The same repository contains a CML workflow that uses DVC to reproduce a full ML pipeline (from data preparation to model evaluation) and then makes training results available on GitHub via the `cml-send-comment` command.

5 Conclusions

In this paper, we present a preliminary investigation of MLOps practices in GitHub. Despite the popularity of the code hosting platform, we highlight a substantial lack of open-source projects concerning ML-enabled systems or components which leverage GITHUB ACTIONS. We argue that such kinds of projects might be hosted as private repositories. Conversely, many ML-related open-source libraries and frameworks leverage GITHUB ACTIONS. However, these were not relevant for the study of MLOps practices as they are no different from traditional software systems, apart from the ML-related goal or use-case. The current query for projects that leverage CML – a specialized MLOps tool – also yielded a few results. However, the tool developers have reached out to us to suggest a refined version of the query, which is likely to improve the recall of CML-based GitHub Actions workflows. Finally, to further overcome the limitations of this study, researchers interested in MLOps practices should take into account further code hosting platforms (e.g., GitLab and Bitbucket), CI/CD services (e.g., TravisCI and CircleCI), and specialized MLOps tools. For more details, please refer to the full version of the paper (Calefato et al., 2022).

References

Amine Barrak, Ellis E. Eghan, and Bram Adams. 2021. On the Co-evolution of ML Pipelines and Source Code - Empirical Study of DVC Projects. In *2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, Honolulu, HI, USA, 422–433. https://doi.org/10.1109/SANER50967.2021.00046

Sumon Biswas, Md Johirul Islam, Yijia Huang, and Hridesh Rajan. 2019. Boa Meets Python: A Boa Dataset of Data Science Software in Python Language. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE, Montreal, QC, Canada, 577–581. https://doi.org/10.1109/MSR.2019.00086

Fabio Calefato, Filippo Lanubile, and Luigi Quaranta. 2022. A Preliminary Investigation of MLOps Practices in GitHub. In *ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. ACM, Helsinki Finland, 283–288. https://doi.org/10.1145/3544902.3546636

Christof Ebert, Gorka Gallardo, Josune Hernantes, and Nicolas Serrano. 2016. DevOps. *Ieee Software* 33, 3 (2016), 94–100. Publisher: IEEE.

Timothy Kinsman, Mairieli Wessel, Marco A. Gerosa, and Christoph Treude. 2021. How Do Software Developers Use GitHub Actions to Automate Their Workflows?. In *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*. IEEE, Madrid, Spain, 420–431. https://doi.org/10.1109/MSR52588.2021.00054

Grace A. Lewis, Stephany Bellomo, and Ipek Ozkaya. 2021. Characterizing and Detecting Mismatch in Machine-Learning-Enabled Systems. In *2021 IEEE/ACM 1st Workshop on AI Engineering - Software Engineering for AI (WAIN)*. IEEE, Madrid, Spain. https://doi.org/10.1109/WAIN52551.2021.00028

Thomas A. Limoncelli. 2018. GitOps: A Path to More Self-Service IT: IaC + PR = GitOps. *Queue* 16, 3 (June 2018), 13–26. https://doi.org/10.1145/3236386.3237207 Place: New York, NY, USA Publisher: Association for Computing Machinery.

Sasu Makinen, Henrik Skogstrom, Eero Laaksonen, and Tommi Mikkonen. 2021. Who Needs MLOps: What Data Scientists Seek to Accomplish and How Can MLOps Help?. In *2021 IEEE/ACM 1st Workshop on AI Engineering - Software Engineering for AI (WAIN)*. IEEE, Madrid, Spain, 109–112. https://doi.org/10.1109/WAIN52551.2021.00024

Nadia Nahar, Shurui Zhou, Grace Lewis, and Christian Kästner. 2021. Collaboration Challenges in Building ML-Enabled Systems: Communication, Documentation, Engineering, and Process. *arXiv:2110.10234 [cs]* (Dec. 2021). http://arxiv.org/abs/2110.10234 arXiv: 2110.10234.

Ipek Ozkaya. 2020. What is really different in engineering ai-enabled systems? *IEEE Software* 37, 4 (2020), 3–6. Publisher: IEEE.

Mark Treveil, Nicolas Omont, Clément Stenac, Kenji Lefevre, Du Phan, Joachim Zentici, Adrien Lavoillotte, Makoto Miyazaki, and Lynn Heidmann. 2020. *Introducing MLOps: how to scale machine learning in the enterprise*. O'Reilly Media, Inc. https://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=2696044 OCLC: 1226320327.

Appendix A

Definition of GitHub Actions *actions* found in the workflows from the second experimental dataset:

- actions/checkout Clones a repository in the virtual environment in which a job is running;
- iterative/setup-cml Enables CML functionalities within GitHub Actions workflows.
- actions/setup-python Specifies the version of Python to be used in the next steps of a job.

- `iterative/setup-dvc` Sets up DVC functionalities within GITHUB ACTIONS workflows.
- `ros-industrial/industrial_ci` Configures a CI process specifically tailored for packages powered by ROS (Robot Operating System).
- `aws-actions/configure-aws-credentials` Configures AWS credentials and region environment variables to be used for AWS API calls.

Appendix B

Definition of CML *commands* found in the workflows from the second experimental dataset:

- `send-comment` adds a Markdown report as a comment on a commit or pull/merge request.
- `publish` uploads and publicly hosts an image (e.g., a plot) for displaying in a CML report.
- `runner` starts a runner in any supported on-premise or cloud computing provider.
- `tensorboard-dev` returns a link to tensorboard.dev, a managed TensorBoard platform that lets users upload and share their ML experiment results.