Demo Abstract: Scanflow: An end-to-end Agent-based Autonomic ML Workflow Manager for Clusters

Peini Liu
Barcelona Supercomputing Center
Universitat Politècnica de Catalunya
Barcelona, Spain
peini.liu@bsc.es

Gusseppe Bravo-Rocca
Barcelona Supercomputing Center
Universitat Politècnica de Catalunya
Barcelona, Spain
gusseppe.bravo@bsc.es

Jordi Guitart
Barcelona Supercomputing Center
Universitat Politècnica de Catalunya
Barcelona, Spain
jordi.guitart@bsc.es

Ajay Dholakia
Lenovo Infrastructure Solutions Group, Lenovo
Morrisville, NC, USA
adholakia@lenovo.com

David Ellison
Lenovo Infrastructure Solutions Group, Lenovo
Morrisville, NC, USA
dellison@lenovo.com

Miroslav Hodak
Lenovo Infrastructure Solutions Group, Lenovo
Morrisville, NC, USA
mhodak@lenovo.com

ABSTRACT

Machine Learning (ML) is more than just training models, the whole life-cycle must be considered. Once deployed, a ML model needs to be constantly managed, supervised and debugged to guarantee its availability, validity and robustness in dynamic contexts. This demonstration presents an agent-based ML workflow manager so-called Scanflow[1], which enables autonomic management and supervision of the end-to-end life-cycle of ML workflows on distributed clusters. The case study on a MNIST project[2] shows that different teams can collaborate using Scanflow within a ML project at different phases, and the effectiveness of agents to maintain the model accuracy and throughput of the model serving while running in production.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Multi-agent systems**; **Planning and scheduling**; • **Computer systems organization** → **Self-organizing autonomic computing**.

KEYWORDS

Scanflow, Machine Learning Workflow, Autonomic, Agent, Kubernetes

ACM Reference Format:
Peini Liu, Gusseppe Bravo-Rocca, Jordi Guitart, Ajay Dholakia, David Ellison, and Miroslav Hodak. 2021. Demo Abstract: Scanflow: An end-to-end Agent-based Autonomic ML Workflow Manager for Clusters. In *International Middleware Conference Demos and Posters (Middleware '21 Demos and Posters), December 6–10, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3491086.3492468

[1]https://github.com/bsc-scanflow/scanflow
[2]https://github.com/bsc-scanflow/scanflow/tree/main/tutorials/mnist

Middleware '21 Demos and Posters, December 6–10, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-9154-2/21/12.
https://doi.org/10.1145/3491086.3492468

1 INTRODUCTION

Machine Learning (ML) approaches have become common with good results in different tasks such as machine translation, image classification, recommendation systems, and speech recognition. While working on a ML project, ML workflows composed of some reproducible steps and executed as a pipeline are widely used to build or deploy a model efficiently because of the flexibility, portability and fast delivery they provide to a ML life-cycle.

ML workflows still face several challenges while being used by different teams. The Data Science team requires to automate some repetitive tasks within ML workflows while training and improving the model. Therefore, some AutoML modules and frameworks [5] have been developed to tune hyper-parameters in order to have good learning performance with less human assistance. However, ML life-cycle is more than just training a model [1]. Once the model has been trained, the Data Engineer team works on deploying the ML workflows into production. More importantly, they are required to operate the workflows to maintain the robustness of the model, such as to deal with security vulnerabilities, concept drift, lack of explainability and interpretability, and hidden technical debt. Also, the online inference model serving services have strict latency and efficiency requirements that should be considered. Therefore, ML workflows are no longer running in a known context and with static requirements, meaning that how to enable the autonomy to manage ML workflows has become an open issue [2].

The AutoML techniques proposed in previous works are turned off after training a model, thus cannot help the model to meet dynamic changes after being deployed. To make an autonomic system for ML in production, Kedziora et al. [2] and Zliobaite et al. [6] provided conceptual level frameworks for autonomous adaptive systems, identifying their characteristics and challenges, but without any practical implementation or evaluation. Seldon [4] provides a set of tools for deploying ML models at scale which include practical oversight and governance for ML models. But they mainly focus on monitoring metrics, providing model explanations, and detecting outliers and drift, rather than autonomically maintaining model performance under those circumstances.

Currently, there is not any extensible framework bringing autonomy for ML workflows in production. Therefore, our work enables

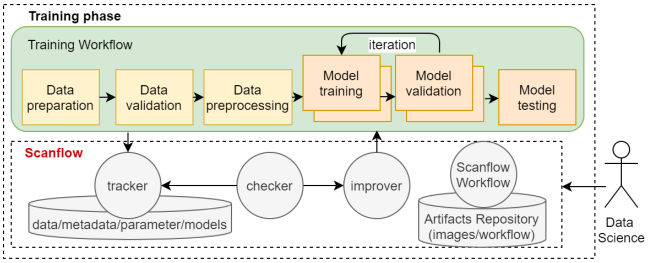

Figure 1: Data Science team works at training phase

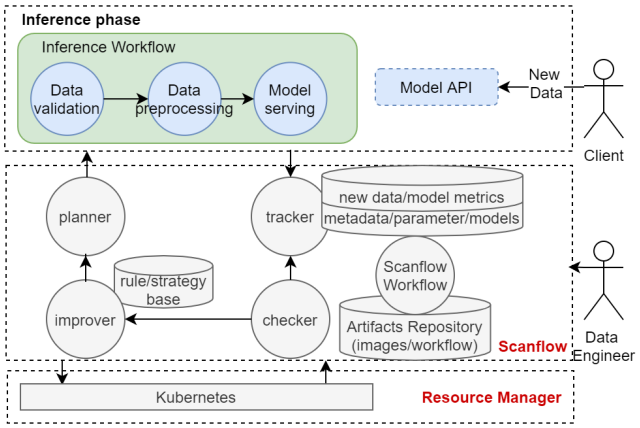

Figure 2: Data Engineer team works at inference phase

an agent-based approach so that ML workflow systems can benefit from autonomic computing to meet dynamic changes. The agents focus on the robustness and requirements of the model at the ML application layer as well as reasoning quality of services and the structure of workflows at the infrastructure layer. For that purpose, we contribute Scanflow[1], an agent-based framework that enables autonomic management of the end-to-end life-cycle of ML workflows on distributed clusters.

2 RESEARCH AND TECHNICAL APPROACH

Scanflow overview: Overall, Scanflow supports defining different types of workflows and building a run-time environment for each step wrapped as a container image. Figure 1 describes the steps for the Data Science team to develop a model. In this phase, Scanflow can track the metadata (such as metrics and scores) and the artifacts, support their analysis through the Checker agent, and automatically tune the hyper-parameters through the Improver agent.

Figure 2 describes the steps for the Data Engineer team to put the ML model into production. First, the deployment of the model inference workflow, which could be either in batch or online mode. The latter will wrap and deploy the model as a serving service. Second, the autonomic operation of the workflow from both the ML application layer and the infrastructure layer thanks to the Scanflow integration with Kubernetes [3]. From the application layer, Scanflow (i.e, agents) can track the model metrics (such as scores) and artifacts (such as new data observations), detect outliers, data drift, provide model explanations, and finally trigger updates of the ML workflow (such as retraining the model with the new data). From the infrastructure layer, we can take profit from the resource management capabilities of the orchestrator. In particular, the quality of the model serving service (such as the latency and failure rate of invocations) can be monitored in real-time. With these observations, Scanflow can collaborate with the resource manager in order to autonomically scale out the service instances or change their resource allocation to improve the reliability and throughput of the model serving.

Agent collaboration for model debugging: Scanflow internally supports four templates of agents, namely Tracker, Checker, Improver, and Planner. The Data Engineer team can provide custom functions to enhance the capabilities of each agent and deal with different robustness problems. As a proof-of-concept, Algorithm 1 outlines the interaction and collaboration of built-in Scanflow agents which feature a non-trivial drift anomaly detector that autonomically deals with out-of-distribution samples in the data and improves a target accuracy estimator based on human feedback to label new data.

Algorithm 1: Agent-based *model debugging*

Input: tracker-agent; checker-agent; improver-agent; planner-agent; *newdata*: new predictions samples; m: current model; q: current model accuracy;
while *size(newdata)* > 1000 **do**
  tracker-agent(*newdata*) call checker-agent;
  $anomaly, picked \leftarrow$ checker-agent(*newdata*);
  while *size(picked)* > 100 **do**
    checker-agent(*picked*) call improver-agent;
    $m', q' \leftarrow$ improver-agent(*picked*);
    if $q' > q$ **then**
      improver-agent(m', q') call planner-agent; m replaced by $m' \leftarrow$ planner-agent(m')
    end
  end
end

ACKNOWLEDGMENTS

This work was partially supported by Lenovo as part of Lenovo-BSC 2020 collaboration agreement, by the Spanish Government under contract PID2019-107255GB-C22, and by the Generalitat de Catalunya under contract 2017-SGR-1414 and under grant 2020 FI-B 00257.

REFERENCES

[1] Google Cloud. 2021. MLOps: Continuous delivery and automation pipelines in ML. https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning
[2] David Jacob Kedziora, Katarzyna Musial, and Bogdan Gabrys. 2020. AutonoML: Towards an Integrated Framework for Autonomous Machine Learning. arXiv:2012.12600 [cs.LG]
[3] Kubernetes. 2021. Production-Grade Container Orchestration - Automated container deployment, scaling, and management. https://kubernetes.io/
[4] Seldon. 2021. ML deployment for enterprise. https://www.seldon.io/
[5] Quanming Yao, Mengshuo Wang, Yuqiang Chen, Wenyuan Dai, Yu-Feng Li, Wei-Wei Tu, Qiang Yang, and Yang Yu. 2019. Taking Human out of Learning Applications: A Survey on Automated Machine Learning. arXiv:1810.13306 [cs.AI]
[6] Indre Zliobaite, Albert Bifet, Mohamed Gaber, Bogdan Gabrys, Joao Gama, Leandro Minku, and Katarzyna Musial. 2012. Next Challenges for Adaptive Learning Systems. *SIGKDD Explor. Newsl.* 14, 1 (Dec. 2012), 48–55. https://doi.org/10.1145/2408736.2408746