A Multi-Agent System for Automated Machine Learning

Demonstration Track

Bruno Fernandes
University of Minho
Braga, Portugal
bruno.fernandes@algoritmi.uminho.pt

Paulo Novais
University of Minho
Braga, Portugal
pjon@di.uminho.pt

Cesar Analide
University of Minho
Braga, Portugal
analide@di.uminho.pt

ABSTRACT

Machine Learning (ML) focuses on giving machines the ability to forecast, predict, or classify without being explicitly programmed to do so. To achieve such goals, large amounts of data are used to conceive models that can adapt to unseen data and to new scenarios. However, applying ML models to real-world business domains is a resource-intensive and time-consuming effort. Automated machine learning (AutoML) emerged as a way to ease such processes. With this in mind, this study introduces a multi-agent system (MAS) that autonomously go through the entire ML pipeline, with different entities being responsible for the data collection process, for pre-processing the data, and for deploying the best candidate ML model. The conceived MAS is currently implemented in a real-world setting, addressing important societal challenges raised by big urban centers. The obtained results show that this solution proved to be beneficial not only for the data collection and pre-processing tasks, but also for the automated execution of ML models.

KEYWORDS

Automated Machine Learning; Multi-Agent Systems; Smart Cities

ACM Reference Format:
Bruno Fernandes, Paulo Novais, and Cesar Analide. 2022. A Multi-Agent System for Automated Machine Learning: Demonstration Track. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 3 pages.

1 INTRODUCTION

Machine Learning (ML) has been rising to prominence. This sub-domain of Artificial Intelligence focuses on giving machines the ability to classify, predict, or forecast, without being explicitly programmed to do so, being able to address complex non-deterministic problems. Several distinct stakeholders are currently using ML. Examples include the use of ML for time-series forecasting [6, 10], medical imaging [3, 7], boredom and stress detection [2, 4], and quantum physics [1, 9]. This diversity brought the need for an automated way to implement and conduct ML processes. In fact, the typical ML pipeline consists in a set of steps that go from data collection to model training and deployment. With this in mind, automated machine learning (AutoML) can be understood as the automation of the ML pipeline, i.e., the use of automated processes for the application of ML models to real-world problems.

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Online. © 2022 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

AutoML is growing to become a hot topic in both industry and academia [8]. In fact, multiple surveys have studied, benchmarked, and analyzed the current state-of-the-art on AutoML solutions [8, 11]. However, to the best of our knowledge, this manuscript presents the first solution to make use of software agents to automate multiple processes within the ML pipeline. Our multi-agent system (MAS) is comprised of a set of agents that are able to autonomously capture data, pre-process them, and make them available for ML models, which are also embedded by such agents. The goal is to have a fully-autonomous ML pipeline.

The conceived MAS is currently implemented in a real-world setting, being used by a mobile solution to display daily traffic flow and ultraviolet forecasts to citizens of big urban centers [5]. The MAS hosts two main entities, entitled as *The Collector* and *ML Architect*, working twenty-four hours seven days a week. This demonstration focuses on the ability of such a MAS to autonomously gather and collect significant data from a city's environment and to go through the entire ML pipeline, making use of the best candidate models to provide accurate forecasts, as described in the next lines.

2 THE COLLECTOR

A software agent, entitled as *The Collector*, was conceived to autonomously collect data from a set of public APIs. This agent is the one responsible for creating and feeding constantly growing datasets. It went live on July 24th 2018 and has been collecting data uninterruptedly. *The Collector* monitors a set of cities and their environmental parameters. It is a modular and configurable agent that can, at any time, start collecting data for any city worldwide. It was developed using Java as programming language, being remotely deployed in an Ubuntu 18.04 machine. Its data collection service is open-source, being available on GitHub, under an Apache License (https://github.com/brunofmf/SmartCity-API).

At the time of writing, four distinct API clients are being queried. One, the *Pollution Client*, uses Open Weather Maps and Open Air Quality APIs. From the first, the agent obtains data about air pollution, including ozone and ultraviolet indexes as well as carbon monoxide, sulfur, and nitrogen dioxide ones. From the second, it extracts data such as black carbon, and particulate matters 2.5 and 10, among others. On the other hand, the *Weather Client* uses Open Weather Maps API to obtain data about the weather including, among others, the weather description, temperature, humidity, atmospheric pressure, wind speed, and cloudiness. It also obtains three days of weather forecasts.

The *Traffic Flow Client* uses TomTom Traffic Flow API to obtain data about the flow of vehicles in several roads of the sensed cities. As features it includes the road identification, the functional road class describing the road type, the current speed in km/h at the observed road, the free-flow speed in km/h expected under ideal free-flow conditions, the current travel time in seconds, and the travel time in seconds that would be expected under ideal conditions, among others. The *Traffic Incidents Client* uses TomTom Traffic Incidents API to obtain data about road incidents for the same cities. As incidents, we may consider jams, accidents, or closed roads, among others. As features, one may find the incident description, its category, the magnitude of the delay it is causing, the cause of the incident, the delay in seconds, and the length in meters of the traffic jam, among others.

As soon as *The Collector* starts, it autonomously collects data and dumps it to a database shared with other agents. *The Collector* makes API calls, using HTTP Get requests, every 120 minutes for pollution APIs, every 60 minutes for weather APIs, and every 20 minutes for the traffic ones. These timings are the default ones and all are runtime configurable. *The Collector* then parses the received JSON object and stores it in a MySQL database (version 14.14 and distribution 5.7.29) using an InnoDB engine (version 5.7.29). Currently, the database holds tens of millions of records, which are used to train and conceive state-of-the-art ML models.

3 ML ARCHITECT

The *ML Architect* is the agent responsible for pre-processing all data in order to guarantee that it is ready to be fed to the ML models. It is also the one responsible for updating, training, and using ML models to provide accurate forecasts and predictions. An hourly job executes the deployed models and disseminates predictions and forecasts. For that, it uses real-time data, gathered by *The Collector*, to obtain predictions and forecasts. This agent was conceived using Python, version 3.7, and is remotely deployed in the same Ubuntu 18.04 machine as *The Collector*. Figure 1 provides a high-level view of the ML pipeline implemented by the conceived agents.

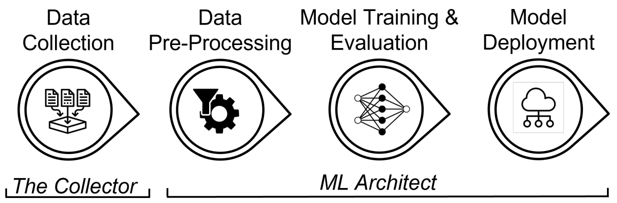

Figure 1: High-level view of the ML pipeline implemented by each agent

As soon as the *ML Architect* intends to use a ML model, its first behavior is to obtain the required data. Then, it autonomously prepares the data as required by the model it aims to use. It then uses the most up-to-date pre-trained model to obtain forecasts, dumping the obtained values to a cloud platform. In fact, the *ML Architect* follows a publisher-subscriber pattern, i.e., after executing the ML models and obtaining the respective forecasts, the model's output is then committed to a feed that is subscribed by other agents. Currently, the *ML Architect* is using Firebase's Cloud Firestore, a flexible and scalable database for mobile, web, and server development from Google's Cloud Platform. Cloud Firestore is a cloud-hosted, NoSQL database, with a data model that supports flexible and hierarchical data structures, with data being stored in documents that contain fields mapping values. Documents are, in turn, stored in collections, which are documents' containers used to organize data and build queries. Documents support several data types, including strings, numbers, and complex nested objects. Any other agent, or software, that subscribes to *ML Architect*'s feeds, gets access to daily forecasts. That is the case of SafeCity, where each user of the mobile application assumes the *persona* of an agent, subscribing to *ML Architect*'s feeds and, thus, accessing forecasts and predictions made by this agent.

4 DEMO

This manuscript's demonstration focuses on the conception and use of a MAS for automated ML. The MAS holds agents that autonomously gather city's environmental data and that are responsible for fulfilling the entire ML pipeline. While *The Collector* is constantly gathering data for multiple cities, the *ML Architect* is using ML models to provide accurate forecasts, following a publisher-subscriber pattern. In particular, the *ML Architect* currently holds and implements state-of-the-art deep learning models to forecast the traffic flow at multiple cities as well as their ultraviolet index. *SafeCity* subscribes to *ML Architect*'s feeds, thus providing actionable forecasts to its users. Figure 2 depicts the main views of SafeCity, in particular the one presenting traffic flow forecasts based on *ML Architect*'s forecasts. A demonstration video is available at https://youtu.be/bC8mNBKsNXA.

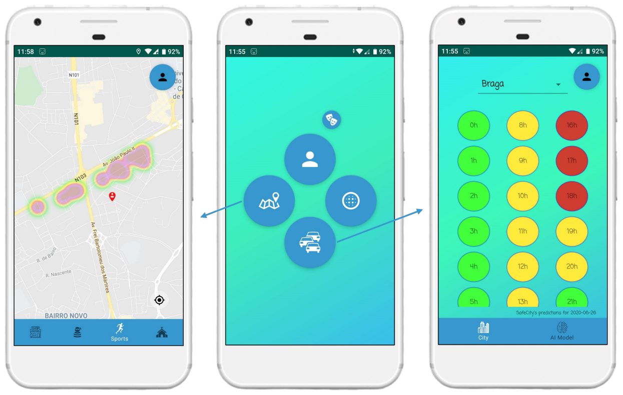

Figure 2: Views of SafeCity's mobile application, presenting traffic flow forecasts based on *ML Architect*'s forecasts

Overall, the MAS implements an automated ML service that has been running for several months uninterruptedly. It not only has been collecting and creating new datasets that keep growing constantly, but it also deploys ML models that are used to obtain daily forecasts. In particular, *SafeCity* uses the conceived MAS to obtain traffic flow and ultraviolet index forecasts. In addition, *SafeCity* implements a traffic alert service, which warns users, every morning, about high traffic hours for their city for the day.

ACKNOWLEDGMENTS

This work has been financed by FCT – *Fundação para a Ciência e Tecnologia* within the scope of project DSAIPA/AI/0099/2019.

REFERENCES

[1] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. 2017. Quantum machine learning. *Nature* 549 (2017), 195–202. https://doi.org/10.1038/nature23474

[2] Franca Delmastro, Flavio Di Martino, and Cristina Dolciotti. 2020. Cognitive Training and Stress Detection in MCI Frail Older People Through Wearable Sensors and Machine Learning. *IEEE Access* 8 (2020), 65573–65590. https://doi.org/10.1109/ACCESS.2020.2985301

[3] Bradley J. Erickson, Panagiotis Korfiatis, Zeynettin Akkus, and Timothy L. Kline. 2017. Machine Learning for Medical Imaging. *RadioGraphics* 37, 2 (2017). https://doi.org/10.1148/rg.2017160130

[4] Bruno Fernandes, Carlos Campos, José Neves, and Cesar Analide. 2020. A Machine Learning Approach to Boredom Detection using Smartphone's Sensors. In *9th European Starting AI Researchers' Symposium (STAIRS) co-located with the 24th European Conference on Artificial Intelligence (ECAI 2020)*. ECAI, 8. http://ceur-ws.org/Vol-2655/paper10.pdf

[5] Bruno Fernandes, José Neves, and Cesar Analide. 2020. SafeCity: A platform for Safer and Smarter Cities. In *Advances in Practical Applications of Agents, Multi-Agent Systems, and Trustworthiness (PAAMS 2020)*, Vol. 12092. 412–416. https://doi.org/10.1007/978-3-030-49778-1_37

[6] Bruno Fernandes, Fábio Silva, Hector Alaiz-Moretón, Paulo Novais, José Neves, and Cesar Analide. 2020. Long Short-Term Memory Networks for Traffic Flow Forecasting: Exploring Input Variables, Time Frames and Multi-Step Approaches. *INFORMATICA* 31, 4 (2020), 723–749. https://doi.org/10.15388/20-INFOR431

[7] Eli Gibson, Wenqi Li, Carole Sudre, Lucas Fidon, Dzhoshkun I. Shakir, Guotai Wang, Zach Eaton-Rosen, Robert Gray, Tom Doel, Yipeng Hu, Tom Whyntie, Parashkev Nachev, Marc Modat, Dean C. Barratt, Sébastien Ourselin, M. Jorge Cardoso, and Tom Vercauteren. 2018. NiftyNet: a deep-learning platform for medical imaging. *Computer Methods and Programs in Biomedicine* 158 (2018), 113–122. https://doi.org/10.1016/j.cmpb.2018.01.025

[8] Xin He, Kaiyong Zhao, and Xiaowen Chu. 2021. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems* 212 (2021), 106622. https://doi.org/10.1016/j.knosys.2020.106622

[9] Hsin-Yuan Huang, Michael Broughton, Masoud Mohseni, Ryan Babbush, Sergio Boixo, Hartmut Neven, and Jarrod R. McClean. 2021. Power of data in quantum machine learning. *Nature Communications* 12 (2021), 2631. https://doi.org/10.1038/s41467-021-22539-9

[10] Reihane Rahimilarki, Zhiwei Gao, Nanlin Jin, and Aihua Zhang. 2019. Time-series Deep Learning Fault Detection with the Application of Wind Turbine Benchmark. In *IEEE 17th International Conference on Industrial Informatics (INDIN)*. 1337–1342. https://doi.org/10.1109/INDIN41052.2019.8972237

[11] Quanming Yao, Mengshuo Wang, Yuqiang Chen, Wenyuan Dai, Yu-Feng Li, Wei-Wei Tu, Qiang Yang, and Yang Yu. 2018. Taking Human out of Learning Applications: A Survey on Automated Machine Learning. *arXiv e-prints* (2018). arXiv:1810.13306 [cs.AI] https://ui.adsabs.harvard.edu/abs/2018arXiv181013306Y