Research Reports on Computer Science
https://ojs.wiserpub.com/index.php/RRCS/

UNIVERSAL WISER PUBLISHER

Research Article in Special Issue: Selected Papers from the 4th International Conference on Machine Learning, Image Processing, Network Security and Data Sciences (MIND-2022)

MLOps for Enhancing the Accuracy of Machine Learning Models using DevOps, Continuous Integration, and Continuous Deployment

Medisetti Yashwanth Sai Krishna*, Suresh Kumar Gawre

Department of Electrical Engineering, Maulana Azad National Institute of Technology, Bhopal, India
E-mail: yashwanthsaikrishna@gmail.com

Received: 8 March 2023; **Accepted:** 14 April 2023

Abstract: Machine learning (ML) integrated with development and operations (DevOps) is the key to solving the problem of deploying the latest machine learning models. This paper proposes one of the ways of integrating machine learning with DevOps. The need for this integration is endless as this provides seamless upgradation of the so-created models while also making managing and monitoring simple. The paper also provides light on practices of Continuous Integration/Continuous Deployment (CI/CD) and minimizing the unnecessary loss of time while training an ML model. The procedure followed includes CI/CD that contains jobs to train the models and to roll out the model with maximum performance. The main focus of this paper is the dynamic change of hyperparameters to achieve increased accuracy without the necessity of the physical presence of humans to change it. This research is independent of the type of machine learning model used and can be best followed for neural networks.

***Keywords*:** ML, CI/CD, machine learning operations (MLOps), DevOps, automated machine learning (AutoML), neural networks

1. Introduction

Machine learning (ML) finds its use in almost every sector in today's world and is recommended for improved performance of the company. The industry has inculcated this tech into its business logic and has almost become the core of many businesses that run around the world. The only drawback of ML is the huge lumps of data that are required to train the models. As data is the one that keeps changing, data is the one that makes models outdated [1]. For a successful business, it is essential to roll out the best and most advanced services, so these ML models have to be up-to-date with the data and deployed in the production systems. The process of deploying systems with software is the job of the development and operations (DevOps) teams and continuous delivery of ML models is what it takes to solve the problem of deploying ML models seamlessly. This process is called machine learning operations (MLOps) [2]. While MLOps is more inclined toward deploying the models into servers, the research focuses on improving the efficiency of models by dynamically changing the parameters precisely through various DevOps practices rather than vaguely changing the hyperparameters inside the models which can be time-consuming [3]. It is statistically observed

DOI: https://doi.org/10.37256/rrcs.2320232644

that a significant time can be saved while doing so, also solving the challenge of hyperparameter optimization in parallel [4]. This procedure involves training the ML models inside the docker containers and monitoring the model's accuracy through a Continuous Integration/Continuous Deployment (CI/CD) tool called Jenkins. The rest of the paper is structured as follows: Section 2 includes the motivation whereas Section 3 discusses the procedure with subsections that elaborate on the exact use of each tool in the process. Finally, on a closing note, Sections 4 and 5 include the result analysis and conclusion respectively.

2. Literature survey

It is a well-known fact that 90% of ML models never hit the market as the creation of these models has some serious drawbacks which include unpredictable results and the requirement of highly advanced computing systems to do the necessary calculations required [5]. While the problem of computation can be solved by Big Data tools like Hadoop, the problem of unpredictable results remains. The reason is that the models take days just for the training purpose and the wait time does not guarantee satisfactory results. This usually turns out to be a waste of time and resources for the company. The current situation with machine learning is that companies can't just invest money in data scientists and ML engineers, which most likely results in obsolete models. In other words, only one in ten of an ML engineer's workdays end up producing something useful for the company [6]. Automating this wait time process is the main point of concern and this automation can lead to the actual deployment of the 90% of ML models that never saw existence. The research papers on MLOps discuss the theoretical approach to achieving integration and mostly focus on different fields where MLOps can be implemented. Table 1 contains the research done in specified fields.

Table 1. Research done in specified fields

Method	Advantages	Disadvantages
ML – Convolutional Neural Networks (CNN) [7-10]	1. Self-organization. 2. No formal programming is required. 3. Ability to adapt and learn, fault tolerance, pattern recognition, intuition, prediction, and statistical pattern reconstruction.	1. Not recommended if precise answers are required. 2. Requires greater computational capacity. 3. Cannot justify answers and offline training is difficult and sometimes very tedious.
DevOps [5, 11]	1. Can be implemented by automating the performance testing of web servers. 2. Increased velocity of software delivery. 3. Improve the quality, collaboration, and efficiency of the software.	1. Requires strong communication and collaboration among multiple teams. 2. Integrating DevOps into an environment without compromising security or performance can be challenging. 3. Requires expertise in multiple tools and technologies whose licenses are expensive.
MLOps [2-4]	1. Kubeflow allows multi-user support to simplify collaboration and access management. Katib can be utilized for automated hyperparameter adjustment. 2. Amazon SageMaker can be connected to data sources of the cloud, like Amazon Simple Storage Service (Amazon S3); Athena and Relational Database Service (RDS) also allow verification of each prediction.	1. Monitoring the effectiveness of ML models and poor understanding of solution needs. 2. More inclination of developers towards manual testing and deployment of ML models. 3. Silos between the development and production teams.

3. Methodology

To train the models with multiple sets of hyperparameters automatically, a fully connected automation system should be in place [7]. An overview of constructing this pipeline is mentioned below.

(i) With a docker container image created that consists of the processed data and the model, the first step is to train it with default hyperparameters.

(ii) When trained, check for accuracy to find if further training is required.

(iii) After vigorous training with multiple hyperparameters, deploy the model to production if the desired accuracy is met.

Figure 1 gives an overview of the processes that run inside CI/CD to update the ML models.

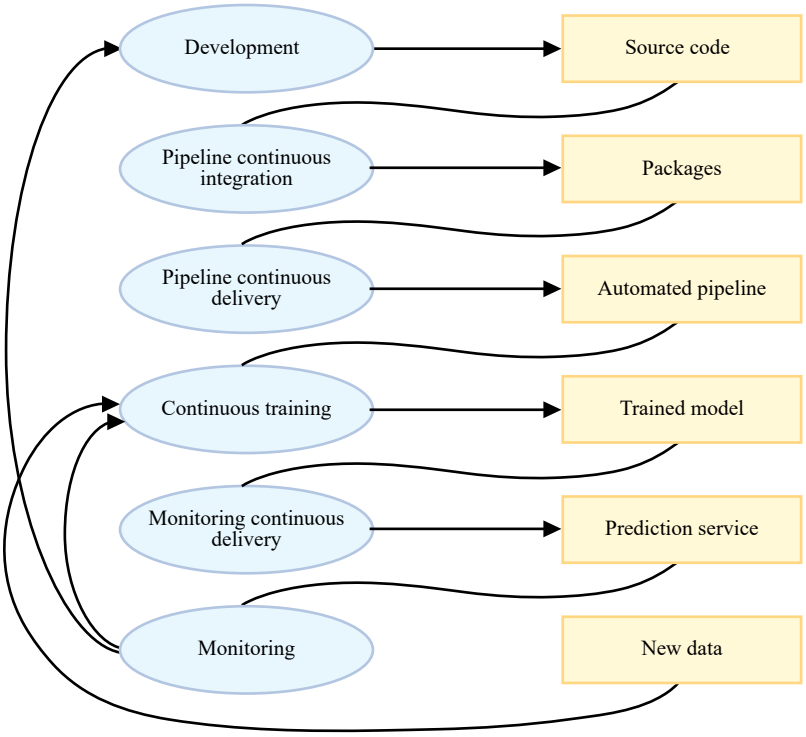

Figure 1. Schematic flowchart of the build pipeline

3.1 *Creating a CNN model*

The procedure is compatible with almost every neural network, a CNN is only considered an example. CNN primarily deals with the extraction of different patterns (or) features within various images for a specific task of image recognition [8]. The kernel (or) filter program is responsible for this process of feature extraction by screening the images. Hence, making CNN a better method for complex pattern and image recognition tasks [9]. A CNN is a combination of multiple layers that include convolutional layers, max-pooling layers, a flattened layer, and a fully connected feed-forward network. This architecture remains highly independent from model to model [10]. A convolutional layer deals with feature selection inside an image, thanks to its kernel program. The max-pooling layers are associated with the process of dimensionality reduction. These layers mostly go in couples and their number itself is a hyperparameter. The flattened layer is responsible for converting the pictorial data into 0s and 1s that can fit inside the fully connected network where the actual training happens [12]. One of the CNN architectures that can be used is found in Figure 2.

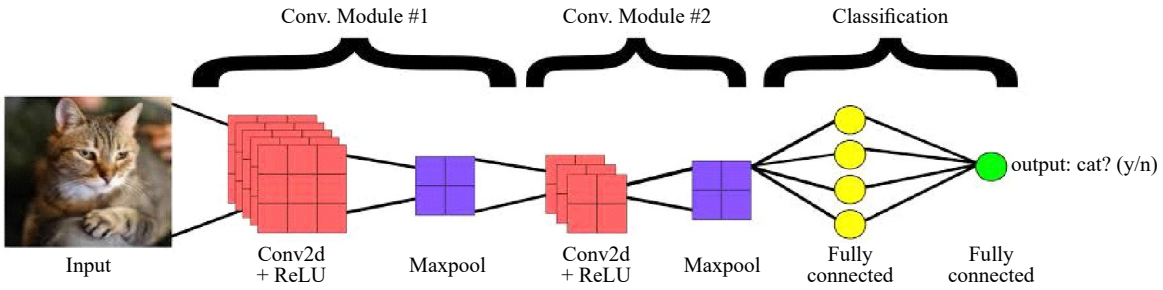

Figure 2. The architecture of a CNN used in the prediction matic flowchart of the build pipeline

The CNN model used here is a simple classifier that recognizes dogs and cats inside an image. This model consists of 2 couples of convolutional modules, a flattened layer that is connected with a neural network consisting of 4 hidden layers and an output layer. There are many hyperparameters in this model which include the number of convolutional and max-pooling layers, activation functions, optimizers, learning rate, neurons used in each hidden layer, loss function, and kernel size. These hyperparameters can be varied to achieve higher accuracy [13]. Approximately 8,000 images have been fitted in the model for the training purpose and 2,000 images to test the created model. A total of 3,359,962 features have been found by the CNN architecture created. The architecture of the CNN model used is specified in Table 2.

Table 2. The architecture of the CNN model used

Model: "Sequential"		
Layer (type)	**Output shape**	**Parameters**
Conv2d	(None, 62, 62, 64)	1,792
MaxPooling2d	(None, 31, 31, 64)	0
Conv2d	(None, 29, 29, 32)	18,464
MaxPooling2d	(None, 14, 14, 32)	0
Flatten	(None, 6272)	0
Dense	(None, 512)	3,211,776
Dense 1	(None, 218)	111,834
Dense 2	(None, 64)	14,016
Dense 3	(None, 32)	2,080
Total parameters: 3,359,962		
Trainable parameters: 3,359,962		

3.2 *ML inside Docker containers*

Training ML models requires maximum utilization of computational resources (central processing unit, CPU and random-access memory, RAM) which can be done by isolating these processes. Dockers provide isolation through container applications. Docker is an open-source container-based platform developed to create and run applications directly in containerized environments. This process is done quickly and easily by the docker as their deployment time is minimal [14]. Due to the limitation and other resource management strategies, Docker containers use their maximum resources towards executing the program assigned to them [15]. The main libraries and software required to run an ML script inside a container are mostly Python, TensorFlow, Keras, and other modules used in the code. For the sample model, all the required software is inscribed in a docker image which can be accessible on Docker Hub as "yashwanth3/ml-basic". Upon training the created CNN model inside the containers, the model reached an accuracy of 52.43%.

3.3 *Hyperparameter optimization*

Automated machine learning (AutoML) is the field of study that deals with hyperparameter tuning [16]. This process involves the research of the best-fit set of hyperparameters for a particular ML model which can be highly tiring, iterative, and time-consuming. AutoML makes it easy for data scientists to create higher efficient models with minimal effort comparatively. Thus, the introduction of AutoML tools in a company can significantly increase the efficiency of the work of data scientists.

3.4 *Continuous Integration (CI) and Continuous Deployment (CD)*

DevOps is the process of developing software solutions. The two main concepts of DevOps include CI and CD. While CI's main aim is to bring diverse departments of information technology (IT) under a single tree, CD ensures the automatic deployment of software in production. CI/CD plays a major role in the complete procedure from building the container image to precisely training the model [11]. Several tools in the market implement CI/CD which includes Jenkins, CircleCI, TeamCity, Bamboo, etc.

3.5 *Jenkins*

First introduced in 2004 as Hudson, Jenkins is an open-source automation tool that provides an interface to automate almost all developmental processes of IT [17]. Jenkins gains importance mainly through its feature of supporting a myriad number of third-party extensions called "plugins" which are of the least importance in our procedure, while GitHub and Email are the only useful plugins in the process that correspondingly act as bridges in integrating the Source Control Management (SCM) and the mailing server. Jenkins is assigned four jobs that complete the process and are interconnected to trigger the next job as soon as the preceding one is successful. The functionality of every job is explained in detail:

(i) *Git import:* The first job is triggered using the remote Uniform Resource Locator (URL) maneuver. This job works to clone the SCM repository into a server. This repo contains the data and the model that has to be trained. The data is already pre-processed using various feature engineering techniques and can be suitably fed to the model.

(ii) *Build Dockerfile:* Taking the pre-created image that has fulfilled software requirements as the base image, this job uploads the code and data into a container and rebuilds the image using "docker cp" and "docker commit" commands. The image created can start training the model as soon as the container gets launched. Also, the job saves the output into a file in parallel for future reference.

(iii) *Parameter change:* This job checks the accuracy of the model created and exits with a status code "0" if the desired accuracy is achieved, else it changes the hyperparameters inside the code through a file handling command of Linux called "sed". To demonstrate the automatic change in the code, hyperparameters like epochs and kernel size are only changed. Change in these doesn't essentially increase the accuracy but advanced concepts of AutoML can be implemented here to drive the model towards achieving better accuracy every time this job gets triggered.

(iv) *Build changed parameters:* When triggered, this job builds the image which contains the changed parameters. After training is done, job(iii) can be retriggered if no improvement is found in the model's accuracy, else the model is ready for production. To demonstrate the actual production, a mailing server can be connected to the pipeline that rolls out a mail to the operations team when the model is ready for real-world use.

4. Result analysis

Upon vigorous training of the model by the pipeline which took approximately 2 hours, the pipeline produced an improved accuracy of 67.48% for the model. It has to be noted that there is no human intervention at any stage of the pipeline and CI/CD alone is responsible for this automation. While accuracy depends on the set of hyperparameters used, my research presents a whole new way to automate the training wherein the procedures of AutoML can be implemented to optimize the accuracy, eliminating the unwanted wastage of time in the deployment process. As the model is dependent on the data and has to be updated timely, the model architecture also needs to be changed. This customization can be automatically done by the CI/CD pipeline to roll out the best model as and when required.

A detailed table of updated architectures can be found in Table 3. This research is a way to showcase the automation that can be achieved while training complex ML models.

Table 3. Updated CNN architectures by the CI/CD pipeline

CNN architecture	Manual training		Training in CI/CD pipeline	
	Architecture	Accuracy	Architecture	Accuracy
LeNet-5	Convolutional layers: 3	64.78%	Convolutional layers: 6	76.54%
	Fully connected layers: 2		Fully connected layers: 3	
	Filter size: 5×5		Filter size: 3×3	
Alexnet	Convolutional layers: 5	85.67%	Convolutional layers: 5	91.4%
	Fully connected layers: 3		Fully connected layers: 4	
	Filter size: 3×3		Filter size: 2×2	
VGG16	Convolutional layers: 13	71.3%	Convolutional layers: 8	76.45%
	Fully connected layers: 3		Fully connected layers: 5	
	Filter size: 3×3		Filter size: 3×3	

5. Conclusion

Comparing the customized architectures created by the pipeline to the standard ones, an increase of 11.76%, 5.73%, and 5.15% in accuracy can be found in LeNet-5, Alexnet, and VGG16 architectures respectively. The power to seamlessly deliver machine learning software is possessed by MLOps and is rapidly becoming a requirement for companies that implement ML in their business. The paper discusses only the incremental procedures of precision while feature engineering and feed forwarding techniques can be applied to the pipeline to make it a complete end-to-end automated MLOps pipeline that can be enterprise-ready. The research aims to save time for data scientists and ML engineers who can think of innovative ideas and research while their ML models keep updating to become better. This practice also leads to fruitful utilization of the company's resources when implemented effectively thus giving time for the analysts to focus on the business rather than being bothered about uncertain results of the technology. Furthermore, the academic space has focused vigorously on ML model building but too little on operating complex ML systems in real-world scenarios which include monitoring, upgrading, and managing the ML models. As companies compete in rolling out the latest applications, this research especially focuses on saving time while deploying ML models.

Conflict of interest

The authors declare that there is no conflict of interest.

References

[1] Garg S, Pundir P, Rathee G, Gupta PK, Garg S, Ahlawat S. On Continuous Integration/Continuous Delivery for automated deployment of machine learning models using MLOps. In: *2021 IEEE Fourth International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. Laguna Hills, USA: IEEE; 2021. p.25-28. https://doi.org/10.1109/AIKE52691.2021.00010

[2] Mäkinen S, Skogström H, Laaksonen E, Mikkonen T. 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)*. Madrid, Spain: IEEE; 2021. p.109-112. https://doi.org/10.1109/WAIN52551.2021.00024

[3] Karamitsos I, Albarhami S, Apostolopoulos C. Applying DevOps practices of continuous automation for machine learning. *Information*. 2020; 11(7): 363. https://doi.org/10.3390/info11070363

[4] Kreuzberger D, Kühl N, Hirschl S. Machine learning operations (MLOps): Overview, definition, and architecture. *IEEE Access*. 2023; 11: 31866-31879. https://doi.org/10.1109/ACCESS.2023.3262138

[5] Gupta S, Bhatia M, Memoria M, Manani P. Prevalence of GitOps, DevOps in Fast CI/CD Cycles. In: *2022*

International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON). Faridabad, India: IEEE; 2022. p.589-596. IEEE. https://doi.org/10.1109/COM-IT-CON54601.2022.9850786

[6] Joury A. *Why 90% of machine learning models never hit the market*. https://thenextweb.com/news/why-most-machine-learning-models-never-hit-market-syndication [Accessed 5th March 2023].

[7] Osman H, Ghafari M, Nierstrasz O. Hyperparameter optimization to improve bug prediction accuracy. In: *2017 IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE)*. Klagenfurt, Austria: IEEE; 2017. p.33-38. https://doi.org/10.1109/MALTESQUE.2017.7882014

[8] Ashin A, Rathika PD, Mahavidhya Y, Hemapriya N, Gowri SS. Image Classification in the Era of Deep Learning. In: *2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA)*. Coimbatore, India: IEEE; 2021. pp.1-5. https://doi.org/10.1109/ICAECA52838.2021.9675614

[9] O'Shea K, Nash R. An introduction to convolutional neural networks. *arXiv* [Preprint] 2015. Version 2. https://doi.org/10.48550/arXiv.1511.08458

[10] Rampasek L, Goldenberg A. TensorFlow: Biology's gateway to deep learning? *Cell Systems*. 2016; 2(1): 12-14. https://doi.org/10.1016/j.cels.2016.01.009

[11] Illingworth WT. Beginner's guide to neural networks. *IEEE Aerospace and Electronic Systems Magazine*. 1989; 4(9): 44-49. https://doi.org/10.1109/62.35668

[12] Xin R, Zhang J, Shao Y. Complex network classification with convolutional neural network. *Tsinghua Science and Technology*. 2020; 25(4): 447-457. https://doi.org/10.26599/TST.2019.9010055

[13] Lingayat A, Badre RR, Gupta AK. Performance evaluation for deploying docker containers on baremetal and virtual machine. In: *2018 3rd International Conference on Communication and Electronics Systems (ICCES)*. Coimbatore, India: IEEE; 2018. p.1019-1023. https://doi.org/10.1109/CESYS.2018.8723998

[14] Preeth EN, Mulerickal FJ, Paul B, Sastri Y. Evaluation of Docker containers based on hardware utilization. In: *2015 International Conference on Control Communication & Computing India (ICCC)*. Trivandrum, India: IEEE; 2015. p.697-700. https://doi.org/10.1109/ICCC.2015.7432984

[15] Patibandla RSML, Srinivas VS, Mohanty SN, Pattanaik CR. Automatic machine learning: An exploratory review. In: *2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*. Noida, India: IEEE; 2021. p.1-9. https://doi.org/10.1109/ICRITO51393.2021.9596483

[16] Pratama MR, Kusumo DS. Implementation of Continuous Integration and Continuous Delivery (CI/CD) on Automatic Performance Testing. In: *2021 9th International Conference on Information and Communication Technology (ICoICT)*. Yogyakarta, Indonesia: IEEE; 2021. p.230-235. https://doi.org/10.1109/ICoICT52021.2021.9527496

[17] Castanier R, Gustafsson L. *How to Test Using Jenkins?* [PDF] 2019. https://fileadmin.cs.lth.se/cs/Education/ETSN20/reports/GroupA.pdf [Accessed 5th March 2023].