International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

Optimizing Efficiency and Performance: Investigating Data Pipelines for Artificial Intelligence Model Development and Practical Applications

Manoj Suryadevara[1], Sandeep Rangineni[2], Srinivas Venkata[3]

[1]Staff Product Manager, Walmart, Arkansas, USA, ORCID: 0009-0007-8738-2222

[2]Data Test Engineer, Pluto TV, California, USA, ORCID: 0009-0003-9623-4062

[3]Staff Data Manager, Teradata, Texas, USA, ORCID: 0009-0007-5547-0383

Abstract: ***Due to the nature of AI, it is difficult for businesses to continually create and deploy models to complicated production systems while maintaining quality. Data processing, model training, code creation, and system management are the pipeline's four steps. We also relate the difficulties of pipeline deployment, modification, and deployment to these four phases of AI evolution. The potential for ongoing model improvement to boost AI performance and flexibility has garnered considerable interest in both academia and industry. This report provides a survey of ongoing efforts in both academia and industry to advance AI model development. We begin with an overview of the pipeline's most crucial parts, which include data collection and preparation, model development and assessment, rollout and monitoring, and iterative refinement. We go into the difficulties at each level and look at recent developments in research and best practices in the field. The next section explores the present status of data collecting and preprocessing studies, with a particular emphasis on methods for gathering and cleaning large-scale datasets, dealing with data bases, and assuring privacy and security. To address the interpretability and fairness of models, we also look at methods for training and evaluating models, such as transfer learning, reinforcement learning, and explainability approaches. We also examine the deployment phase, dissecting the best practices for deploying models across different environments, as well as the advantages and disadvantages of containerization and scalability. We address methods for updating and retraining models, as well as the need of continual monitoring and assessment in detecting model drift, bias, and performance decline. Finally, we examine feedback loops and their function in the continuous development pipeline, with special emphasis on the value of user input, human-in-the-loop strategies, and assessment methods designed with the end user in mind. We talk about the algorithmic bias, transparency, and accountability that are ethical concerns in the ongoing development of AI models. We hope that this in-depth look at the AI model creation process will help academics and practitioners make more informed decisions moving forward. To guarantee the trustworthy and beneficial deployment of AI models across a variety of fields, we address the obstacles and advances at each level, paving the path for future research and highlighting the need for strong and responsible AI development procedures.***

Keywords: Data Pipeline, Artificial Intelligence, Machine Learning Operations, Data Quality

1. Introduction

AI development pipelines are currently the subject of extensive study; synthesizing the existing body of research can help researchers avoid misunderstandings, identify gaps in the field's understanding, and find new avenues for investigation. Important conceptual concepts, consolidate and arrange related research, and promote ongoing progress in the field of pipelines for the creation of AI.

Constant software development and operations (DevOps)

In this portion of the article, we outline the most fundamental prior information that will be necessary. Following a brief overview of CI/CD and DevOps in general, this part provides a more in-depth explanation of these practices as they pertain to artificial intelligence. See Section 4.1 for an explanation of the various AI-related terminology.

Connected Tasks

Several recent works have addressed the topic of AI development pipelines (e.g., Tamburri (2020), Fischer et al. (2020), Renggli et al. (2021), Mäkinen et al. (2021), Mäkinen (2021), Alnafessah et al. (2021), Nguyen-Duc et al. (2020), Kolltveit and Li (2022), Nguyen-Duc Because our research on AI's ongoing evolution follows a similar technique to that of the Systematic Literature Review (SLR) and the Multivocal Literature Review (MLR), we've included a list of relevant prior work below. The table summarizes the aims and methods of similar studies and shows how they connect to our paper's three study goals. The paragraphs that follow then go further into the identified related work and analyze it in detail. Firstly, several related studies (Karamitsos et al., 2020; John et al., 2021a,b; Lwakatare et al., 2020b; Figalist et al., 2020; Fredriksson et al., 2020; Kolltveit and Li, 2022; Kreuzberger et al., 2022; Lorenzoni et al., 2021) are based on a limited amount of

identified primary sources, not covering relevant insights from the wide range of existing literature. Our research was based on a meta-analysis of more than 150 previous studies. Furthermore, other literature evaluations in related work concentrate on a particular application setting, such as edge/cloud/hybrid architectures (John et al., 2021a), ML-based software analytics and business intelligence applications (Figalist et al., 2020), or federated learning (Lo et al., 2021). In contrast, our research considers all AI models, without regard to their unique implementations. Third, related studies investigate a wide variety of research topics that are not directly connected to the definition of words (RQ1), AI pipelines and triggers for initiating the pipeline (RQ2), or pipeline-related difficulties (RQ3).

2. Challenges

Instead than focusing on the processes involved in continuous development, they searched for lifecycle characteristics like traceability, repeatability, guidelines, and transparency. Second, issues with pipeline itself are seldom discussed in the context of related work, which instead tends to concentrate on broader issues with AI research and development. For instance, Nascimento et al. (2020) show how the many complexities involved in creating AI systems are intertwined with the software engineering methods outlined in the SWE- BOK knowledge domains. The issues at the heart of this article, which Kreuzberger et al. (2022) address, are organizational, ML system, and only very superficially operational. Figalist and co.

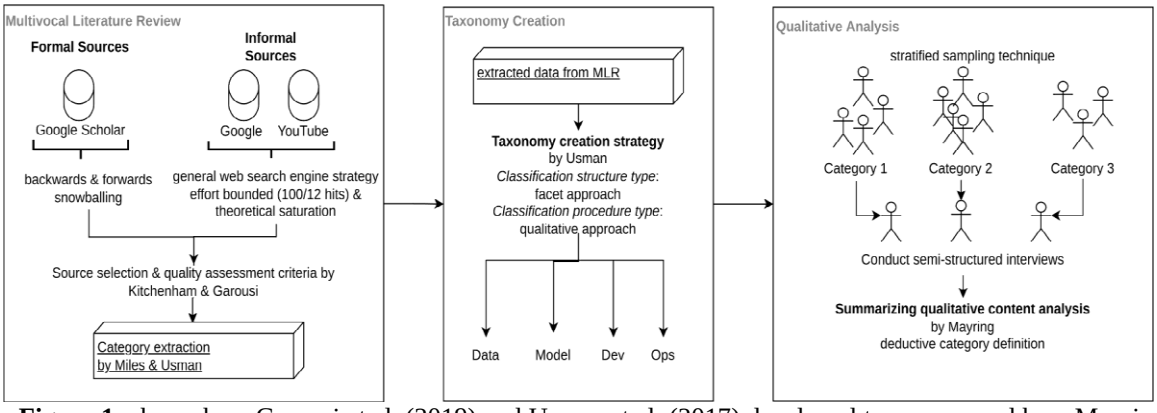


Figure 1: shows how Garousi et al. (2019) and Usman et al. (2017) developed taxonomy and how Mayring (2015) defined deductive categories for use in an MLR study.

(2020) pinpoint difficulties with developing, deploying, and updating, with a special emphasis on AI models for BI and software analytics. Data labeling is only one example of a problem that is unique to artificial intelligence but is not addressed in the paper by Testi et al. (2022). Problems with putting the AI model into practice are the focus of Kolltveit and Li (2022).

Challenges in AI system development are addressed in a related body of work that employs a different methodology than our own research.

3. Methodology of Research

The MLR found a total of 151 relevant sources, 79 of which were papers (53% formal) and 72 were not. Online, you may get a systematic map that details the extraction procedure and the scholarly and popular literature that was culled from it (Steidl et al., 2022).

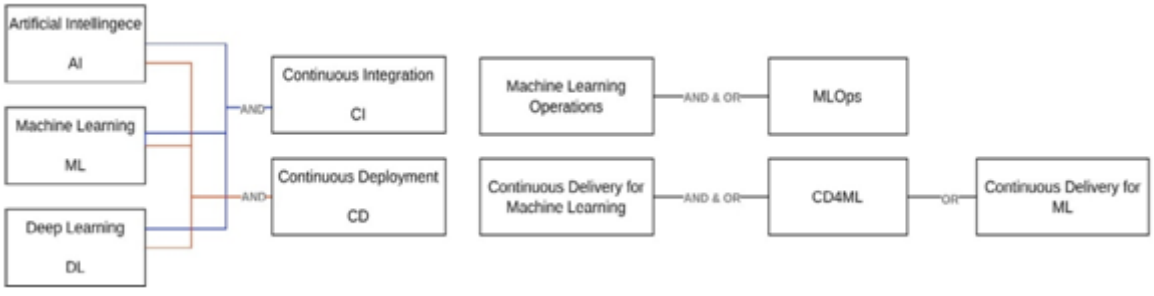


Figure 2: The MLR search phrases used to find relevant articles.

We conducted a descriptive qualitative synthesis to glean the relevant categories from the existing literature. Based on the predetermined research objectives, Kitchenham and Charters (2007) insist that you record the data you collect in a consistent, tabular way. Steidl et al. (2022) has further details on the tabulation of the retrieved data. Following the advice of Stol et al. (2016) and Usman et al. (2017), we break down the research topics into even more specific

categories and develop the taxonomy. To dissect, analyze, compare, interpret, and classify the data, we used the open and axial coding technique proposed by Stol et al. (2016). The pilot research serves as the basis for our classifications. Usman et al. (2017) found that these groups were well accepted since they were similar to those used in continuous software engineering.

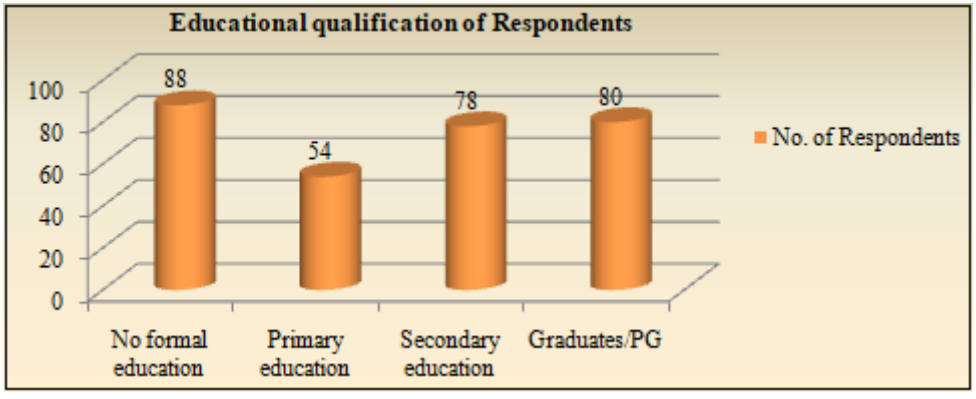
Development of a Taxonomy

Based on the educational background the respondents are classified as having No formal education, having Primary education, secondary Education, Graduates/PG educational background. The respondents were divided into four groups based on their level of education, as shown in the table below.

Table for pipeline in artificial intelligence model systems based on the Educational background

"Education	No. of Respondents	Percent
No formal education	88	29.3
Primary education	54	18
Secondary education	78	26
Graduates/PG	80	26.7
Total	300	100

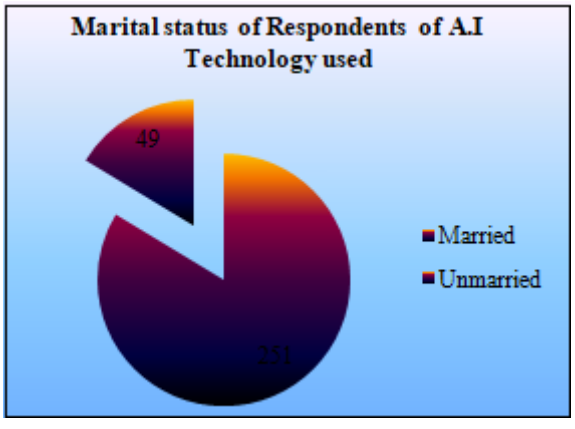
According to the data shown above, 29.3 percent of those surveyed had not completed any formal schooling, 18% of the respondents are having primary level education, 26% are having secondary level education, 26.7% are graduates and postgraduates.



Pipeline utilization in artificial intelligence-based marital status categorization table

Marital Status	No. of Respondents	Percent
Married	251	83.7
Unmarried	49	16.3
Total	300	100

The above shows that out of 300 respondents 251 (83.7%) respondents are married and 49 (16.3%) respondents are Unmarried".

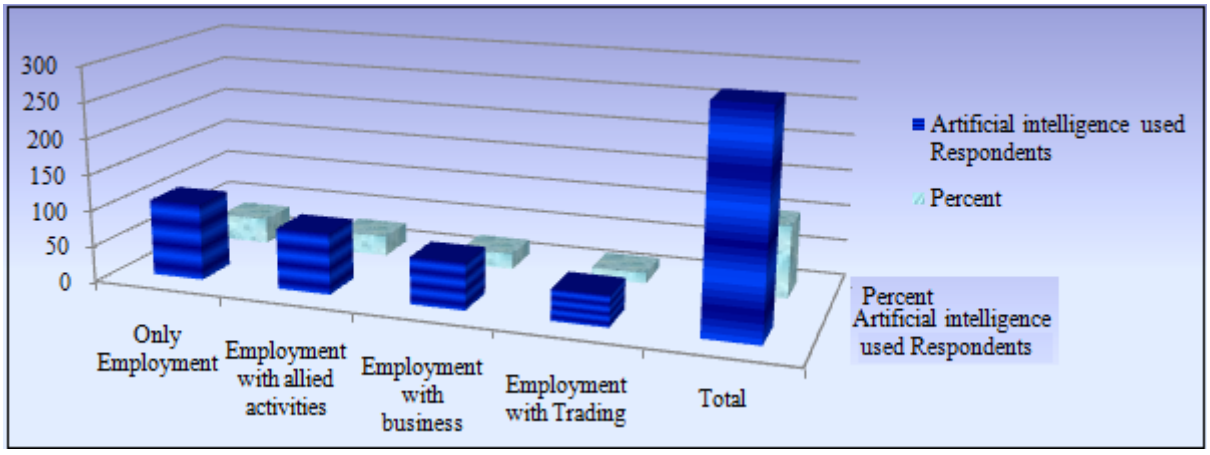


Economic Background

a) Occupation:
Scheduled the foundation of Livelihood performed the respondents are classified into four categories as performing Only Employment, Employment with allied activities, Employment with business and Employment with Trading. "The details of the classification are as given in the below table":

Table: Classification of respondents based on the Occupation

"Occupation	No. of Artificial intelligence used Respondents	Percent
Only Employment	107	35.7
Employment with allied activities	85	28.3
Employment with business	65	21.7
Employment with Trading	43	14.3
Total	300	100"



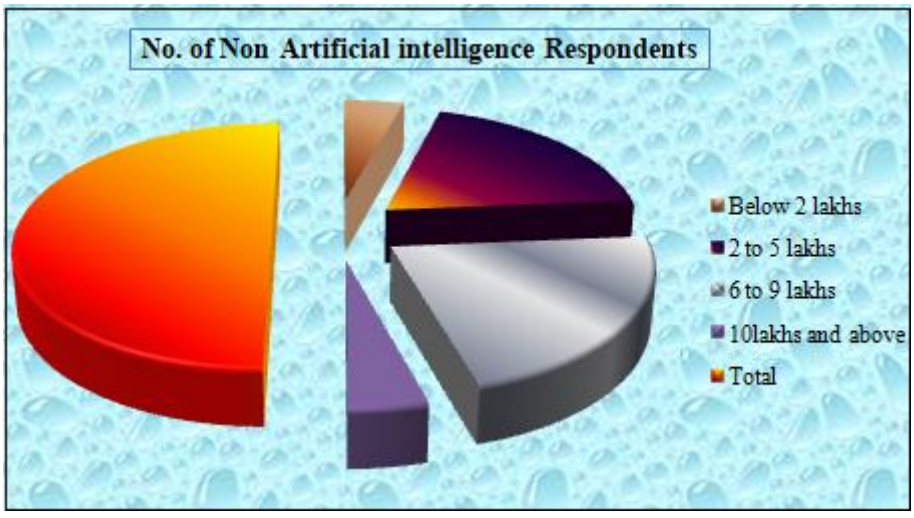
According to the above statistics 35.7 percent of respondents rely only on Employment, 28.3% of respondents are doing Employment along with allied activities, 21.7% of respondents are doing Employment along with business and 14.3% of respondents are doing Employment with Trading as their occupation.

b) Yearly Income:

The respondents' average yearly income are classified into 4 different categories like annual income below 2 lakhs, in between 2 to 5 lakhs, between 6 to 9 lakhs and 10 lakhs and above income per year. The percentages of different occupation of respondents are given below in the table as shown

Table 4.9: Classification of respondents based on the Annual income

"Annual income	No. of Non Artificial intelligence Respondents	Percent
Below 2 lakhs	26	8.7
2 to 5 lakhs	116	38.7
6 to 9 lakhs	132	44
10lakhs and above	26	8.7
Total	300	100



The above table reveals that 44 percent of the Non A.i used respondents had yearly incomes between 6 and 9 lakhs, 38.7 percent have annual incomes between 2 and 5 lakhs, and 8.7 percent of respondents have annual incomes of 10 lakhs or more.

This is investigated by researchers. Data Handling, Model Learning, Software Development, and System Operations are the phases that make up the specified aspects.

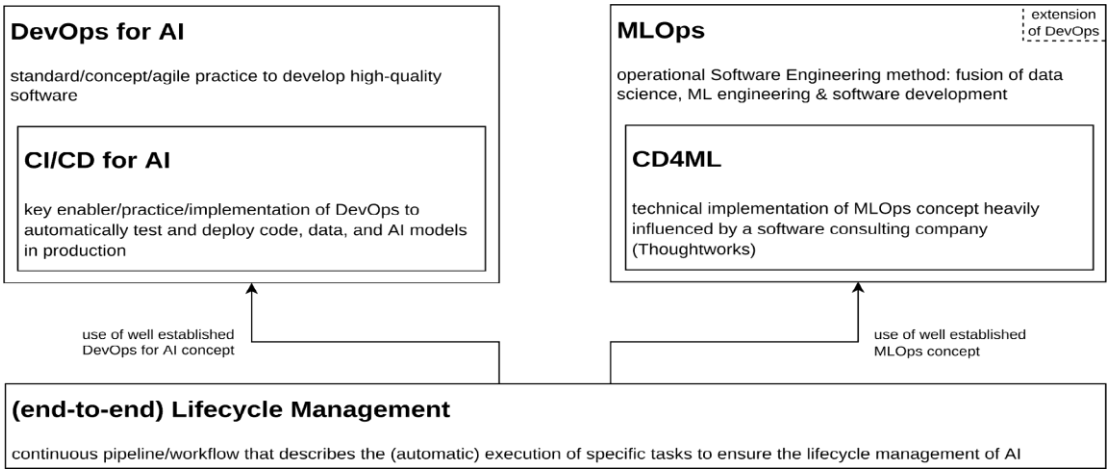


Illustration 3: A glossary of AI-related concepts, including DevOps, CI/CD, MLOps, E2E lifecycle management, and CD4ML.

Pipeline

In this part, we'll go through the pipeline's four steps and the duties performed at each. The first stage, "Data Handling," involves the actual execution of data handling; the second, "Model Learning," involves the actual implementation Our literature review and in-depth interviews allowed us to identify a taxonomy of phases, which is shown in Figure. 4. Companies A, B, C, and V (out of a total of nine) do not completely automate their pipelines.

Figure 4 shows that the pipeline is not a straight line but rather involves several feedback loops. This is an essential aspect of agile development since it enables the incorporation of constant input to enhance. The goal provide a comprehensive overview of all the quality assurance methods that may be used to AI. Only methods that we found during the MLR to be useful for continuous pipelines are shown. Furthermore, there is substantial variation across implementations in terms of the scope and varieties of tests.

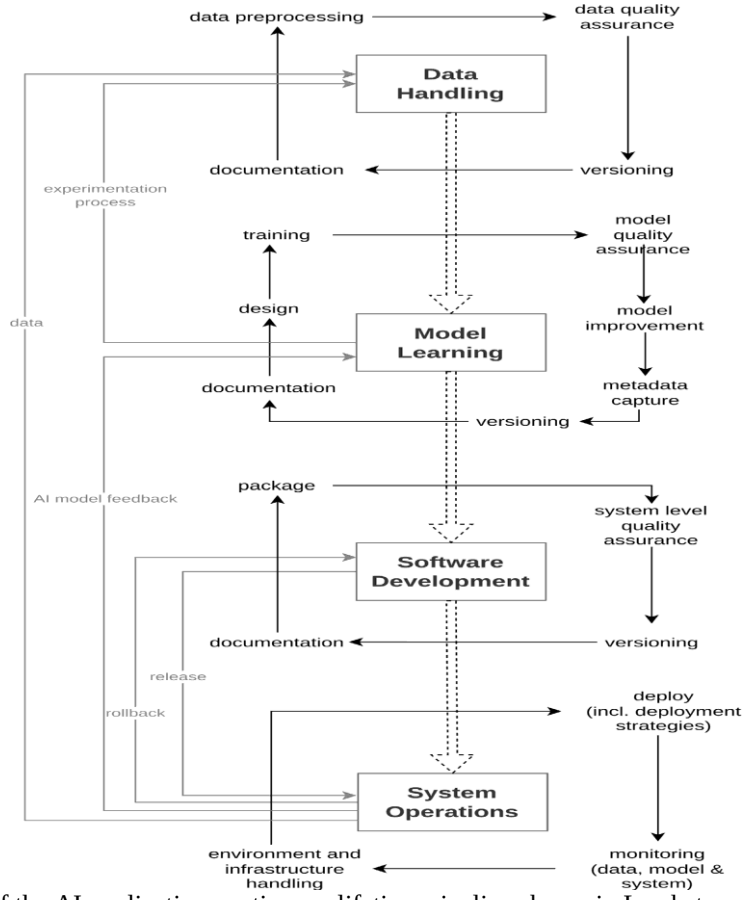


Figure 4 is an adaptation of the AI application continuous lifetime pipeline shown in Lwakatare et al. (2020a) and Tamburri (2020).

Quality Control for Data

Since the pipeline is always running, it's important to include reusable testing components to ensure that no errors are introduced into the data facilitates the early detection of inaccurate input, preventing it from wreaking havoc on the model and wasting computational resources on inaccurate output. By determining the gap in data distribution between the two sets (training and serving), interesting statistical features may be uncovered. To determine whether the difference is significant enough to constitute an error, a distance threshold is determined. The probabilities of the distance are derived from two distributions (Cave- ness et al., 2020). (Breck et al., 2019) TFX incorporates these strategies. As part of the standard validation procedure, A and R double-check that the data is in the correct form. The data collection is unique if and only if it contains no duplicates. The degree to which the semantic norms of a data collection have been broken is shown by the concept of consistency. The validity of data determines whether or not all values are in the correct format. How well the data fits and is approved to carry out a certain job is what we mean by "accuracy." Whether or if a data collection is timely for a given job is described by its "timeli- ness" (Renggli et al., 2021).

Challenges

In this part, we'll go through 25 obstacles that need to be overcome in order to ensure that AI is always improving. All of the problems that have been discovered are plotted against the four phases of the provided framework.The frequency with which each task is mentioned in the sources is shown in Fig. 5 by a distinct color for each challenge. Example: a yellow bar shows that we extracted the challenge less than five times, while a dark blue bar shows that we recognized the problem between twenty-six and thirty times. Table 7 provides a more detailed summary of these difficulties. We also provide a more in-depth description of the difficulties we encountered most often in the next section.

Operations on Data

Problems that arise during the act The complexities of data handling include its collection. Over 16 situations have been cited where data collection and integration have been problematic during the data stage. The pipeline has difficulties in meeting regional data requirements (Banerjee et al., 2020). In the healthcare industry, for instance, it may be mandatory to retain data sets internally.

Learned models

Model quality assurance (MQA) is complicated by specific challenges that arise during the Model Learning stage, which compound the difficulty of knowing when to update a model for maximum efficiency.For this reason, it is important that the pipeline's versions be well-documented so that the root cause of any performance or forecast shifts can be determined (Yun et al., 2020).

Creation of New Software

Participant P found it difficult to address certain difficulties that arose during the Software De- velopment stage, such as those related to the independence of the packaging model and software in order to enable scalability. When building a complicated model, it might be difficult to trace provenance if it incorporates many models that were trained using different data. If the modifications in the training data set are highly-heterogeneous, the problem becomes considerably more difficult (Lwakatare et al., 2019).

Workings of the Scheme

Particular difficulties arising at this juncture Challenges with deployment are discussed in detail by System Operations, including guaranteeing model.Concerns with monitoring center on identifying actionable metrics and uncovering latent feedback loops. Multiple environments and embedded systems were noted as a difficulty during environment and infrastructure management, and this was confirmed by 12 different sources.

Pipeline Requirements in General

The necessity for a versatile, adaptable, re-usable, and fault-tolerant pipeline is at the forefront of addressing general difficulties that cannot be localized to a particular stage of the architecture. Because of fluctuations in capacity, elastic scaling presents a unique difficulty for pipelines. The absence of rules, the complexity of the multi-tenant environment, and the need to integrate with current security systems all present difficulties.

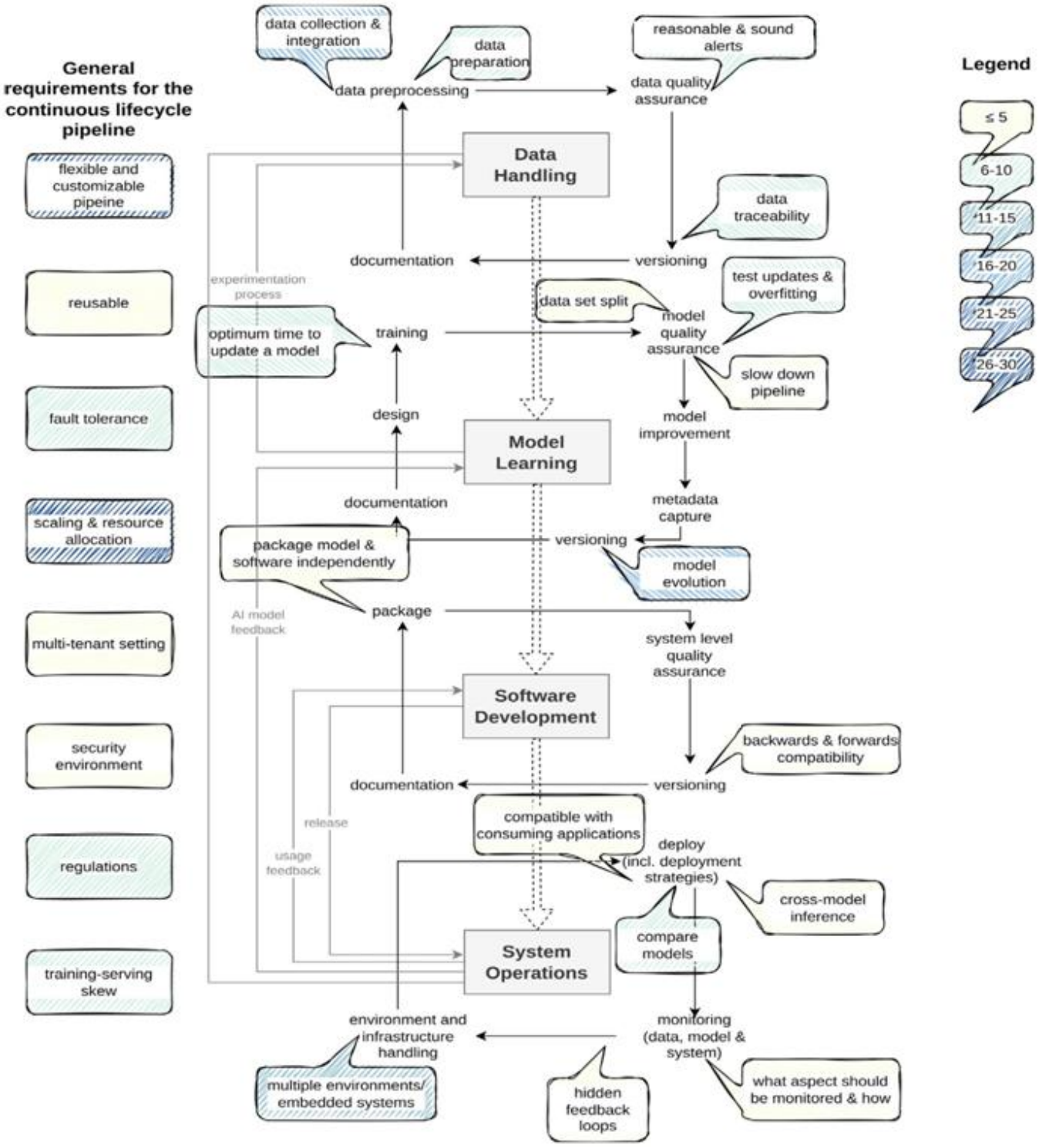


Figure 5: Difficulties encountered in AI pipeline installation, adaptation, and use, colored according to frequency of occurrence .V mentioned that his clients wished to run the TFX pipeline independently but lacked the necessary Linux-based infrastructure to do so. Since the complexity of algorithms is growing at an unexpected rate, the demand for computational resources is very variable. In order to supply the appropriate quantity of hardware to deal with capacity variations, elastic scaling is essential. Both horizontal and vertical scalability are required of the pipeline.

The MLR data shows that TFX is the most often referenced pipeline. To provide this adaptability, TFX integrates many components.

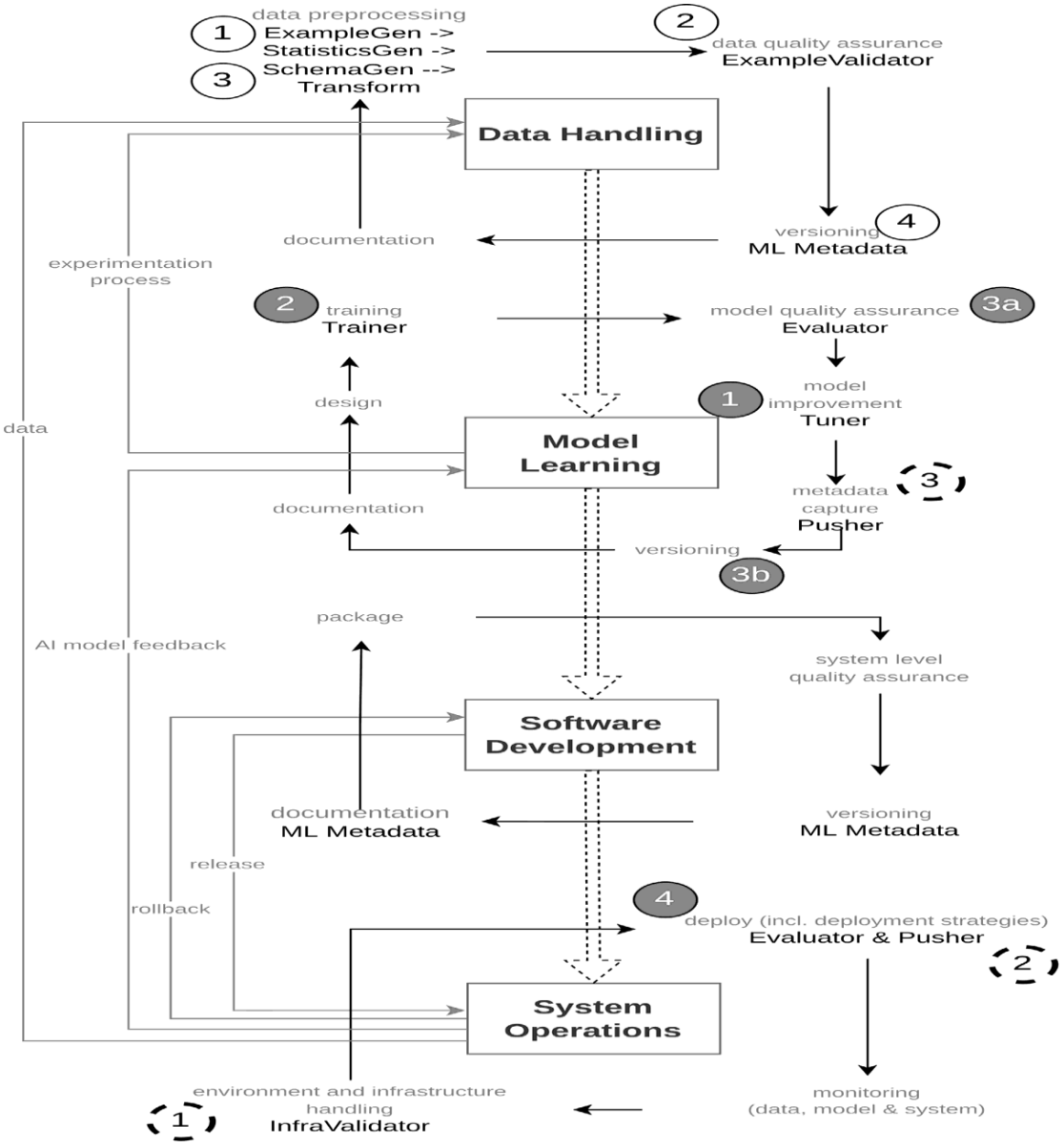


Figure 6: Relate TFX subsystems to steps in the proposed architecture. The numbered circles show the sequence in which each TFX stage is executed, and the empty circle represents the stage we've pinpointed. The unfilled circle represents the data preparation phase, the dotted circle the model learning phase, and the full circle our system operations phase and our adaptable pipeline. These parts are coordinated and triggered by an orchestrator like Kubeflow. Tasks, as used in this study, are synonymous with TFX components within the context of the proposed framework. Fig. 6 shows how the TFX fits within the overall structure. As will be shown in the next section, ExampleGen, StatisticsGen, SchemaGen, and Transform are the components responsible for the preliminary processing of data.

Operations on Data

Data from various file formats (such as csv) is ingested by TFX's ExampleGen component, which then saves the information in a suitable manner for the following components. In the next step, ExampleGen separates the data for each variant into test and development sets. In order to generate descriptive statistics for a dataset, Statistics Gen takes the results from ExampleGen as input.SchemaGen employs these metrics throughout the automatic schema construction process. Data types, intervals, categories, distributions, and other metadata are all described in the schema. The resulting schema is flexible and open to change by the developer. To guarantee high-quality data, Example Validator feeds the created schema into TensorFlow's data validation service. In accordance with the MLR's recommendations, it evaluates a series of data and compares its statistical properties to the produced schema, analyzes training and serving data to spot differences between the two, and so on.

Learned models

In contrast to our suggested approach, TFX optimizes the model by tweaking its hyperparameters before proceeding to train and verify it. For better model learning, TFX employs the Python KerasTuner in their Tuner to fine-tune the model's hyperparameters. Tuner takes as input a collection of data for both training and evaluating, as well as tuning logic like the model specification and the search space for hyperparameters, and a series of Protocol buffers with instructions for serializing structured data.

In this case, the Python TensorFlow API is used to train an AI model. In this step, we train two models using the data set provided by ExampleGen and the modified features. Both models are used for assessment purposes by TFX, but only one is put into production for inference. In addition, TFX enables "warm-starting" the training using an already-existing model. Additionally, the ML Metadata Store allows developers to compare numerous model runs concurrently during model training.

The Evaluator in TFX examines the outcomes of the training and provides a window into the model's behavior for subsets of the data. This guarantees the detection of models that do well across the board but badly on a single data point. The best model for this job may be determined by comparison to others; it need not be the most recently trained model. Furthermore, the accuracy, precision, recall, etc. are all quantified using the normal Keras metrics.

Creation of New Software

Packaging and system-level quality assurance are two software development-specific activities that TFX's basic pipeline can not handle. ML Software-specific metadata, such as version numbers and documentation.

Workings of the System

The metadata repository lets you see which characteristics had an effect on the metrics you used for assessment. To save time, developers may modify the model's hyperparameters instead of running the whole pipeline again.

Threats to validity

Because we used a qualitative method, the reliability of our findings is compromised. We developed and tested an interview strategy with the aim of increasing the trustworthiness of treatment implementation interviews. once first source selection is complete. Thirty papers were selected at random, with just two not being included. There is now a 93.3 percent degree of agreement. In order to classify the data from the sources we use, we compute a test-retest value. Thus, we once again culled data from these 30 diverse resources. On average, there is a 23% discrepancy in the data we're able to retrieve. This may sound like a lot, but consider that in some categories we found two sources during the retest whereas before we'd only found one: a 100% increase!

By merging works by the same authors and publishers, (3) redundant information was eliminated. For example, in (4) the given checks, unnecessary materials that were disregarded. In addition to (6) the search strings allowing for the inclusion of numerous relevant research results, (5) contradictory findings were also included.The companies with whom we conducted interviews are either established players in the AI market or cutting-edge startups working on AI application development and the related lifecycle pipeline. These discussions also served to assess and broaden the MLR's findings, providing a more complete picture of the kinds of work that must be done to ensure AI's ongoing progress.The goal of this research is to provide a thorough structure for the on-going improvement of AI models. This study therefore integrates studies in AI with those in ML and DL. Since the MLR's findings were consistent with each other regardless of the tasks implemented for data handling or model learning, the report did not differentiate between them.

4. Conclusion

The primary objective was to carefully extract the most important theoretical frameworks, as well as to synthesize and organize the existing literature. Potential primary triggers were also explored in the article. These included feedback and alarm systems, timed orchestration services, regular repository updates, and manual triggers. Data Handling encompasses a wide range of iterative operations, including data preparation, quality control, version control, and documentation. Graphically representing the steps used to create a model, such as "model design," "model training," "model quality assurance," "model improvement," "model metadata capture," "model versioning," and "documentation," is what "Model Learning" does with the data once it has been processed. System Operations is phase after deployment when the AI model is put into use in a live setting. The report also includes a mapping from those four stages to twenty-five potential deployment-related difficulties. The difficulties of model learning highlight the need for versioning to show how the model has changed over time and to guarantee its repeatability. The software development pipeline must ensure model and data schema compatibility both in the past and the future. System Operations requires support for several contexts with the potential for cross-model inferences. Finally, the difficulties that arise as a consequence of the pipeline specifications are described in detail. Model DB, Facebook's FB Learner, and Uber's Michelangelo AI lifecycle platform are a few more possible alternatives. This paper's taxonomy of defined tasks may be compared to the existing tasks covered by the aforementioned platforms.

References

[1] Sandeep Rangineni et al. (2023), The Review of Contemporary Scientific and Academic Studies, vol 3, issue 6, https://doi.org/10.55454/rcsas.3.06.2023.002.
[2] Divya Marupaka et al. (2023), Data Pipeline Engineering In The Insurance Industry: A Critical Analysis Of Etl Frameworks, Integration Strategies, And Scalability, IJCRT, Vol 11, Issue 6, http://doi.one/10.1729/Journal.34747.
[3] Sandeep Rangineni et al. (2023), An Examination of Machine Learning in the Process of Data Integration, International Journal of Computer Trends and Technology, Volume 71 Issue 6, 79-85, https://doi.org/10.14445/22312803/IJCTT-V71I6P114
[4] Alnafessah, Ahmad, Gias, AlimUl, Wang, Runan, Zhu, Lulai, Casale, Giu-liano,Filieri,Antonio,2021.Quality-aware DevOps research: Wheredo we stand?IEEEAccess9,44476–44489.http://dx.doi.org/10.1109/ACCESS.2021.3064867.
[5] Baier, Lucas, Jöhren, Fabian, Seebacher, Stefan, 2019. Challenges in the deployment and operation of machine learning in practice.
[6] Boucher, Philip, 2020. Artificial intelligence: How does it work, why does it matter, and what can we do about it?. European Parliament, Brussels.
[7] Dellinger, Amy, 2005.Validity and the review of literature. Res. Schools 12.Ereth,Julian,2018.DataOps-Towards a Definition. In: Proceedings of the Conference Lernen,Wissen,Daten,Analysen.LWDA,URLhttps://ceur-ws.org/Vol-2191/paper13.pdf.
[8] Figalist,Iris,Elsner,Christoph,Bosch, Jan, Olsson, Helena Holmström, 2020.Anend-to-end framework for productive use of machine learning in soft-ware analytics and business intelligence solutions. In: Morisio, Maurizio, Torchiano, Marco, Jedlitschka, Andreas (Eds.), Product-Focused Software Process Improvement. In: LNCS sublibrary, SL Programming and SoftwareEngineering, vol. 12562, Springer, Cham, pp. 217–233. http://dx.doi.org/10.1007/978-3-030-64148-1_14.
[9] Fischer,Lukas, Ehrlinger, Lisa, Geist, Verena, Ramler, Rudolf, Sobiezky, Florian,Zellinger, Werner, Brunner, David, Kumar, Mohit, Moser, Bernhard, 2020. AIsystem engineering key challenges and lessons learned. Mach. Learn. Knowl. Extraction3(1),56–83.
[10] Fitzgerald, Brian, Stol, Klaas-Jan,2017. Continuous software engineering:Aroadmapandagenda.J.Syst.Softw.123,176–189.http://dx.doi.org/10.1016/j.jss.2015.06.063,URLhttps://www.sciencedirect.com/science/article/pii/S0164121215001430.
[11] Garousi, Vahid, Felderer, Michael, Mäntylä, Mika V., 2016. The need for multi-vocal literature reviews in software engineering. In: Proceedings of the 20th International Conferenceon Evaluation and Assessment in Software Engi-neering. EASE, ACM, New York, pp. 1–6. http://dx.doi.org/10.1145/2915970.2916008.
[12] Garousi, Vahid, Felderer, Michael, Mäntylä, Mika V., 2019. Guidelines for including grey literature and conducting multivocal literature reviews in software engineering.Inf.Softw.Technol.106,101–121.http://dx.doi.org/10.1016/j.infsof.2018.09.006, URLhttps://www.sciencedirect.com/science/article/pii/S0950584918301939.
[13] Gmeiner, Johannes, Ramler, Rudolf, Haslinger, Julian,2015. Automated testing in the continuous delivery pipeline: A case study of an online company.In:2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW).IEEE,pp.1–6.
[14] Jalali, Samireh, Wohlin,Claes, 2012. Systematic literature studies. In: Rune-son,Per(Ed.),2012ACM-IEEE International Symposiumon Empirical Software Engineering and Measurement. ESEM, IEEE, Piscataway, NJ, p. 29.http://dx.doi.org/10.1145/2372251.2372257.
[15] John, MeenuMary, Holmström Olsson, Helena, Bosch, Jan, 2021a.ArchitectingAI deployment: A systematic review of state-of-the-art and state-of- practice literature. In: Klotins, Eriks, Wnuk, Krzysztof(Eds.), Software Business. Springer International Publishing, Cham ,pp.14–29.
[16] John, Meenu Mary, Olsson, Helena Holmstrom, Bosch, Jan,2021b.Towards MLOps: A Framework and Maturity Model.In: Proceedings-202147th Euromicro Conference on Software Engineering and Advanced Applications,SEAA 2021. Institute of Electrical and Electronics Engineers Inc., pp. 334–341.http://dx.doi.org/10.1109/SEAA53835.2021.00050.
[17] Jolliffe, Ian, 2005. Principal component analysis. In: Everitt, Brian, How-ell, David C. (Eds.), Encyclopedia of Statistics in Behavioral Science. Wiley,Chichester,http://dx.doi.org/10.1002/0470013192.bsa501.
[18] Karamitsos, Ioannis, Albarhami, Saeed, Apostolopoulos, Charalampos, 2020. Ap-plying DevOps practices of continuous automation for machine learning. Information11(7),363.http://dx.doi.org/10.3390/info11070363.
[19] Kim,Gene, Humble, Jez, Debois, Patrick, Willis, John, Allspaw, John, 2016. The DevOpsh and book:Howto create world-class agility, reliability, & security in technology organizations, First edition IT Revolution Press LLC, Portland OR.
[20] Kitchenham, Barbara Ann, Charters,Stuart,2007.Guidelines for performing systematic literature reviews in software engineering,2.
[21] Kolltveit, Ask Berstad, Li, Jingyue, 2022. Operationalizing Machine Learning Models-A Systematic Literature Review. In: Proceedings-Workshop on Software Engineering for Responsible AI, SE4RAI 2022. Institute of Electricaland Electronics Engineers Inc., pp. 1–8. http://dx.doi.org/10.1145/3526073.3527584.

[22] Kreuzberger, Dominik, Kühl, Niklas, Hirschl, Sebastian, 2022. Machine Learning Operations (MLOps): Overview, Definition, and Architecture.http://dx.doi.org/10.48550/arxiv.2205.02302,arXiv:2205.02302.

[23] Lenarduzzi, Valentina, Lomio, Francesco, Moreschini, Sergio, Taibi, Davide, Tamburri, Damian, 2020.Software quality for AI: Where we are now?.

[24] Lewis, Grace A., Ozkaya, Ipek, Xu, Xiwei, 2021. Software Architecture Challenges for ML Systems. In: Proceedings - 2021 IEEE International Conference on Software Maintenance and Evolution, ICSME 2021. Institute of Electrical and Electronics Engineers Inc., pp. 634–638. http://dx.doi.org/10.1109/ICSME52107.2021.00071.

[25] Lo,SinKit, Lu,Qinghua,Wang, Chen, Paik, HyeYoung,Zhu,Liming,2021.ACMComput.Surv.54(5) , Dr. Naveen Prasadula A Review Of Literature On Research And Practical Applications Of The Data Pipeline For Artificial Intelligence Model Development07574v1.

[26] Martínez-Fernández, Silverio, Bogner, Justus, Franch, Xavier, Oriol, Marc,Siebert,Julien,Trendowicz,Adam, Vollmer, Anna Maria, Wagner, Stefan, 2022. Software Engineering for AI-Based Systems: A Survey. ACM Trans. Softw. Eng. Methodol. (TOSEM) 31 (2), http://dx.doi.org/10.1145/3487043,arXiv:2105.01984. Mayring,Philipp1952-,2015. Qualitative In halts analyse: Grundlagenund Techniken, 12., überarbeitete Auflage Pädagogik, Basel:Beltzand [Grün-wald]: Preselect. media GmbH, Weinheim, URL https://bibsearch.uibk.ac.at/AC12283426.

[27] Mboweni, Tsakani, Masombuka, Themba, Dongmo, Cyrille, 2022. A Systematic Review of Machine Learning DevOps.In: International Conference on Electri-cal, Computer, and Energy Technologies, ICECET2022. Institute of Electrical and Electronics Engineers Inc., http://dx.doi.org/10.1109/ICECET55527.2022.9872968.

[28] Messick, Samuel, 1995. Standards of validity and the validity of standards in performance assessment. Edu. Meas. Issues Pract. 14 (4), 5–8. http://dx.doi.org/10.1111/j.1745-3992.1995.tb00881.x.

[29] Miles, Matthew B., Huberman, A. Michael, Saldaña, Johnny, 2014. Qualitative dataanalysis: A methods sourcebook, Edition 3 Sage, Los Angeles and London and New Delhi and Singapore and Washington DC.

[30] Mishra, Alok, Otaiwi, Ziadoon, 2020.DevOps and software quality: A systematic mapping. Comp.Sci.Rev.38,100308.http://dx.doi.org/10.1016/j.cosrev.2020.100308,URLhttps://www.sciencedirect.com/science/article/pii/S1574013720304081.

[31] Munappy, Aiswarya Raj, Mattos, David Issa, Bosch, Jan, Olsson, Helena Holm-ström,Dakkak,Anas,2020.FromAd-Hocdataanalytics to Data Ops. In: Proceedings-2020IEEE/ ACM International Conference on Software and System Processes, ICSSP 2020, vl. 20. Association for Computing Machinery,Inc,pp.165–174.http://dx.doi.org/10.1145/3379177.3388909.

[32] Nascimento,Elizamary, Nguyen-Duc, Anh, Sundbø, Ingrid, Conte, Tayana,2020. Software engineering for artificial intelligence and machine learn-ing software: A systematic literature review. URL http://arxiv.org/pdf/2011.03751v1.

[33] Ng,Andrew, 2011. Sparseauto encoder. CS294 A Lecture Notes 72 (2011), 1–19.Nguyen-Duc, Anh, Sundbø, Ingrid, Nascimento, Elizamary, Conte, Tayana, Ahmed, Iftekhar, Abrahamsson, Pekka, 2020. A Multiple Case Study of Artificial Intelligent System Development in Industry. In:ACM International Conference Proceeding Series. Association for Computing Machinery, pp.1–10.http://dx.doi.org/10.1145/3383219.3383220.

[34] Paleyes, Andrei, Urma, Raoul-Gabriel, Lawrence, Neil D., 2020. Challenges in Deploying Machine Learning: a Survey of Case Studies, URL http://arxiv.org/pdf/2011.09926v2.

[35] Pieters, Wolter, 2011. Explanation and trust: what to tell the user in security and AI? Ethics Inform.Technol.13(1),53–64. http://dx.doi.org/10.1007/s10676-010-9253-3.

[36] Pivarski, Jim, Bennett, Collin, Grossman, RobertL., 2016. Deploying analytics with the portable format for analytics (PFA). In: Krishnapuram, Balaji,Shah, Mohak, Smola, Alex, Aggarwal, Charu, Shen, Dou, Rastogi, Rajeev (Eds.), KDD2016. Association for Computing Machinery Inc. (ACM), New York, NY,pp.579–588.http://dx.doi.org/10.1145/2939672.2939731.

[37] Rodriguez, Manuel, Piresde Araújo, Luiz Jonatã, Mazzara, Manuel,2020.Good practices for the adoption of Data Opsinthes of twareindustry.J.Phys.Conf.Ser.1694,012032.http://dx.doi.org/10.1088/1742-6596/1694/1/012032.