Automation and Development Effort in Continuous AI Development: A Practitioners' Survey

Monika Steidl*, Valentina Golendukhina†, Michael Felderer‡, and Rudolf Ramler§
*†‡*University of Innsbruck*, Austria
‡*German Aerospace Center (DLR), Institute for Software Technology*, Germany
‡*University of Cologne*, Germany
§*Software Competence Center Hagenberg GmbH*, Austria
ORCID: *0000-0002-3410-7637, †0000-0002-8274-5924,‡0000-0003-3818-4442,§0000-0001-9903-6107

***Abstract*—The widespread adoption of AI-enabled systems and their required continuous development and deployment (MLOps) sparks research interest due to the added intricacy of automatically handling data, code, and the model itself. A better understanding of the stages for the continuous development of AI, namely Data Handling, Model Learning, Software Development, and System Operations, and the respective tasks can help to optimize and improve their effectiveness.**
Thus, this paper explores the degree of automation, development effort, importance, utilization of computing resources, and factors contributing to automation throughout these stages and tasks. We conducted a questionnaire-based global survey to explore these topics by analyzing 150 responses from experienced AI, data, and MLOps engineers.
The results determined that the stage System Operations is mainly automated. Whereas several tasks from the other three stages (e.g., data cleaning, data quality assurance, model design, model improvement, and system level quality assurance) are more often partially automated than automated, and documentation-related tasks are mostly not automated or developed. Participants required the highest development effort for the stage Data Handling. Furthermore, the study reveals a negative correlation between automation and the perceived development effort, whereas the importance of the tasks does not seem to affect automation. 93% of participants consider the availability of computing resources, with model training, data transformation, and data cleaning ranked as the most resource-intensive tasks.

***Index Terms*—MLOps, automation of AI development, effort of AI development, importance of tasks for AI development, computing resource consumption, Machine Learning, Artificial Intelligence, Deep Learning, CD4ML**

I. Introduction

The advance of technology, including the availability of more data and computing power, has opened up numerous possibilities for the use of Artificial Intelligence (AI), along with its subsets of Machine Learning (ML) and Deep Learning (DL), across various industries [4]. To fully leverage the capabilities of AI, it is essential to continuously and rapidly integrate AI models into production systems and monitor them while they are evolving and self-adapting [7], [29]. Thus, the idea of Continuous Integration (CI)/Continuous Deployment (CD) and DevOps (DevOps) is applied to AI to enhance the speed, consistency, and reliability of the AI development process. However, their transformation to AI models sparks research interests due to the added intricacy of code combined with data, model, and a larger system-level complexity [2], [6], [9]. The practices for the continuous development of AI are known as DevOps for AI, CI/CD for AI, Machine Learning Operations (MLOps), (End-to-End) Lifecycle Management, or Continuous Delivery for Machine Learning (CD4ML) [27]. The continuous development pipeline is represented by key stages consisting of `Data Handling`, `Model Learning`, `Software Development`, `System Operations`, each of which comprises a subsequent set of tasks for handling critical steps from data to a deployed AI model [27]. By better understanding these stages and respective tasks, we identify opportunities to optimize them and ultimately improve the efficiency, accuracy, and effectiveness of AI development.

One way to improve the continuous development of AI is by examining the degree of automation present [14], [22]. In addition, the perceived development effort, importance, and resource consumption of tasks throughout the continuous development allow organizations to optimize the pipeline by focusing on the areas with the highest impact and prioritizing research and resources accordingly.

Nevertheless, what tasks are automated and how practitioners perceive their required effort and importance remains uncertain. Furthermore, several tasks, such as model training [31], require many computing resources where automation may help with effective resource management to avoid bottlenecks for highly important tasks. Thus, it is essential to identify which tasks practitioners deem most computing resource-intensive.

This paper elaborates on the following research questions based on the identified research gaps:

- RQ_1 To what degree are the tasks automated in the pipeline for the continuous development of AI?
- RQ_2 How much effort is required to develop a pipeline for the continuous development of AI?
- RQ_3 How important is each task in the pipeline for the continuous development of AI?
- RQ_4 What factors are linked to the adoption of automation?
- RQ_5 What tasks in the pipeline for the continuous development of AI are considered by developers as the most and the least computing resource-intensive?

To answer these questions, we conducted a questionnaire-based survey among practitioners with expertise in the continuous development of AI worldwide. We analyzed 150 complete answers to understand the current state, interconnections, and

This work was supported by the Austrian Research Promotion Agency (FFG) in the frame of the project ConTest [888127]

differences.

The remainder of the paper is structured as follows: Section II reports the related work and introduces the main concepts and definitions of the pipeline for the continuous development of AI. Section III describes the methodology and survey design. In Section IV, we answer the RQ by providing the survey results. Section V interprets the results, highlights future work, and identifies potential threats to validity. Finally, Section VI summarizes the paper's results.

II. Background and Related Work

The following section discusses related work and essential background knowledge. Firstly, we discuss the related work and background information regarding the pipeline for the continuous development of AI. Secondly, we focus on the related work regarding adopting MLOps practices and their implications.

Different publications have employed various methodologies to define and summarize the **tasks involved in the pipeline for the continuous development and deployment of AI**, including case studies conducted by Karlas et al. [15], Amershi et al. [2], and John et al. [13]. As these case studies focus on specific cases, literature reviews have begun to combine available information, accumulating necessary tasks and proposing a taxonomy for a continuous development pipeline for AI, as demonstrated in Steidl et al. [27], John et al. [14], Lwakatare et al. [18], Testi et al. [30], Kreuzberger et al. [17], and Alves et al. [1]. The foundation of this survey relies on Steidl et al.'s work [27], which offers an elaborate description of the pipeline. Figure 1 illustrates the stages of the proposed pipeline, namely `Data Handling`, `Model Learning`, `Software Development`, and `System Operations`, along with their respective tasks. Table I defines and describes these tasks.

Five studies focus on the **adoption of MLOps practices and their implications**. Serban et al. [26] surveyed the degree of adoption of software engineering practices for AI and the respective effects based on different demographic characteristics. Makinen et al. [19] identified challenges for data scientists and how they can profit from MLOps. Amershi et al. [2] executed interviews and a survey at Microsoft to identify processes and practices undertaken to develop AI models. Calefato et al. [5] mined open-source projects on GitHub focusing on GitHub Actions and CML[1] to identify how common workflow automation is in ML-enabled systems. Rahman et al. [21] studied the usage of bots for the automation of development tasks by mining three DL libraries (Keras, PyTorch, and Tensorflow). Zhou et al. [31] executed an experiment where they used CI/CD tools and Kubeflow to measure the time and computing resource consumption of the AI platforms during the pipeline's execution. In contrast, this study looks at the degree of automation, development effort, importance, and computing resources of each specific task regarding the pipeline for the continuous development of AI.

[1]Continuous Machine Learning:https://cml.dev/doc, accessed 03.02.2023

TABLE I
Definition of the pipeline's tasks based on [27]

Task	Definition
`Data Handling`	
Data cleaning	Handling of missing values, outliers, data types, correcting corrupted or inaccurate data
Data transformation	Statistical analysis of the data, bringing it to the suitable format for training, labeling, and feature extraction
Data Quality Assurance (QA)	Data and feature validation, data quality measurements, unit tests
Versioning	Storing data to guarantee traceability and compliance with regulations, also versioning of dependencies, data processing steps, and extracted features (feature stores)
Documentation	Guidelines with concrete actions such as feature cleaning or naming conventions applicable to data files or folders
`Model Learning`	
Design	Model purpose, selection of reusable model components, best suitable algorithm, feature selection, data set to split
Training	The process of training the model with the given data
Model QA	Testing a model based on its quality to provide faultless, reliable, and secure models (e.g., accuracy, precision, recall, convergence, robustness, fairness, etc.)
Model improvement	Context and model-specific improvement techniques (e.g., hyper-parameter optimization, pruning, model hardening)
Versioning (& metadata capture)	Storing version artifacts, code, dependencies, and metadata (e.g., provenance information, execution and deployment of the model, training, training date) to backtrack or reproduce different model versions
Documentation	Documenting a model's purpose and technical decisions (algorithm design and chosen model, achieved results)
`Software Development`	
Packaging	The build process packages the code and model logic into build artifacts which are deployed in production
System-level QA	Should identify the correct behavior of the whole system landscape
Versioning	Packaged models, respective QA results, the pipeline, and its associated tasks are versioned
Documentation	Documentation of necessary information for the software release
`System Operations`	
Deployment	Deployment of a model on different environments based on different deployment strategies (e.g., A/B test, shadow deployment)
Monitoring	Monitoring the AI model in production and collecting the necessary information (data, model performance, traditional software monitoring aspects) to improve over time
Environment and infrastructure	Handling of different environments or operational stages where a model is deployed. Includes varying hardware, operation systems, and software version dependencies

III. Methodology

We chose a descriptive survey design to address the research questions and investigate the current state of AI development following the guidelines of Runeson and Höst [25]. We launched an anonymous online questionnaire survey on LimeSurvey to reach many participants. This section outlines the survey design, participants' selection strategy, and data analysis procedures.

A. Survey design

The questionnaire underwent two stages of design. A sample group completed a pilot study to avoid misinterpretations and identify missing elements. Based on their feedback, we refined the questionnaire. Table II presents the final survey questions

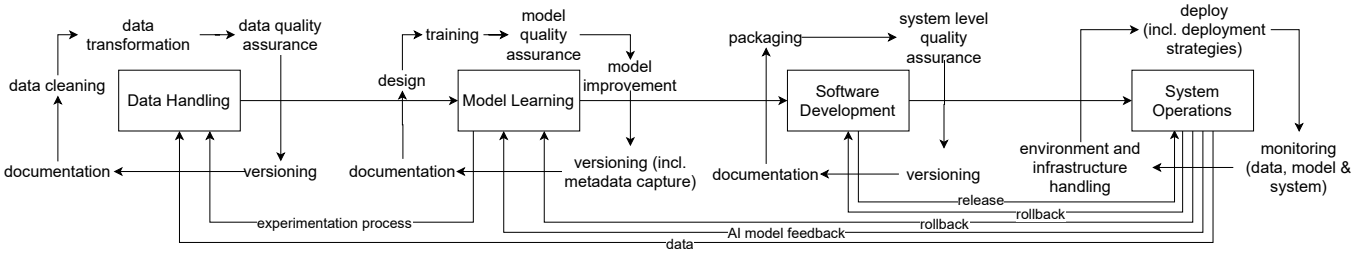

Fig. 1. Pipeline for the continuous development of AI including four stages and respective tasks based on [27]

and refers to the papers from which some question types were adapted.

TABLE II
THE SURVEY QUESTIONS

RQ	Survey Question	Question Type
RQ_1	Do you automate a pipeline task?	4-point ordinal scale (automated, partially automated, not automated, no answer) [14]
RQ_2	Rate the development effort of each task of an ML pipeline Assign 100 points according to the time spent on the development of each stage	4-point ordinal scale (Low, medium, high effort, no answer) assign 100 points to stages
RQ_3	In your opinion, how important is each task?	4-point ordinal scale (not important, low, medium & high importance)
RQ_5	Do you consider computing resources when automating ML tasks? Choose and rank a maximum of 5 of the most resource-intensive phases	yes/no ranking (if participant considers resources)
RQ_4	Experience in AI related field Experience with MLOps Type of data Type of AI models Domain Tools, packages, and libraries used Company size Development team size Gender	Single choice yes/no Multiple choice Multiple choice Multiple choice Multiple choice [24] Single choice Free numerical input Single choice

To facilitate the comprehension and avoid the misinterpretation of the various tasks, each page of the survey contained a graphical representation of the pipeline for the continuous development of AI, including its four stages and 18 tasks as depicted in Figure 1. Additionally, we provided task definitions (see Table I) that participants could access by clicking on the corresponding task.

B. Participants selection

Due to the difficulties in estimating the total number of the target group of AI, data, and MLOps engineers, we used purposive non-probability sampling. Our sample group comprised engineers with the necessary technical expertise to work with and automate AI systems. We reached out to them by publishing a voluntary questionnaire in an AI and MLOps community [2]. Additionally, we distributed the survey to individuals with relevant profiles through LinkedIn and to companies worldwide that develop AI-based products.

Between March and April 2023, a total of 194 responses were received. To proceed with the analysis, we focused on survey participants with more than one year of experience in the researched field who completed the survey. This narrowed our pool to 150 participants with the following profiles.

Over 45% of the participants have 3-5 years of experience in ML-related fields, followed by approx. 23% with 6-10 years of experience and approx. 22% with 1-2 years of experience. 7% of the participants have over ten years of experience. Half of the participants work in large companies with more than 250 employees. The median team size is 10, with an interquartile range of 5 to 30.

The following demographic descriptions are based on multiple-choice questions where participants were allowed to add more than one answer. Most of the participants work in Finance (28%), Healthcare (24%), and E-Commerce (approx. 23%). The majority of the participants work with tabular data (70%), text (60%), and images (55%). Regarding the models they developed, Artificial Neural Network (ANN) were used by approx. 77% of the participants, followed by 59% tree-based models, 58% regression models, and 50% using clustering. The top tool choices were Jupyter Lab, MLFlow, Gitlab, Kubeflow, and AWS SageMaker.

C. Data analysis

To analyze the survey data, we used a combination of descriptive statistics and statistical testing. We provide the survey questions, raw survey data, and complete analysis online [28]. Firstly, we analyzed how participants perceive tasks and stages in terms of automation, development effort, importance, and computing resource consumption. Participants who completed at least half of the questions were included in the analysis, but those who skipped complete sections were excluded from the analysis of these sections. Secondly, we explored the relationships between automation, effort, and importance across tasks, stages, and the overall pipeline. We aggregated responses for each stage and calculated the overall pipeline correlation based on 18 tasks. Missing values were eliminated, resulting in varying numbers of observations. The number of observations per task ranged from 73 to 108, from 258 to 594 per stage, and overall 1600-1684 observations were analyzed. The data and model stages had the most complete responses, while development and operation had more incomplete answers. To estimate the degree of correlation, we applied a non-parametric

[2]https://mlops.community/, accessed 04.07.2023

Spearman's correlation coefficient due to the ordinal scale of the data. We only consider correlations with a significance level better than 1%.

IV. Results

In the following section, we present the most noteworthy findings from the survey analysis and illustrate the takeaways in grey boxes at the end of each section.

A. RQ_1: Degree of tasks automation

Figure 2 presents the automation of tasks within each stage. During the `Data Handling` stage, fewer participants automate the data cleaning process than partially automate it. We observed the same trend for data QA. Data versioning is automatically handled in half of the participants' answers, while data documentation is not automatically handled by half of the participants and is not answered (indicating that this task is not developed) by 25%.

During the `Model Learning` stage, over half of the practitioners do not automate the model design; only 9% automate, and 27% partially automate this process. Model improvement tasks are less often automated (19%) than partially automated 41%. Model versioning is automatically handled in over half of the participants' answers, while no automation of the model documentation is prevalent, and 17% are not developing this task at all.

In the `Software Development` stage, the system level QA is mostly partially automated by 32%, whereas complete automation is achieved by 26%. However, nearly the same number of participants indicated that they did not develop this task. Versioning is automatically handled in over half of the participants' answers. Documentation is again not automated for 38% of the time and not developed for 20%.

During the `System Operations` stage, all three tasks are mostly automated. However, these tasks are not developed/not answered in approx. 14% of the time.

- Data cleaning, data QA, model design, model improvement, system level QA are more often partially automated rather than fully automated
- Tasks in the stage `System Operations` are mostly automated
- Over half of the participants indicate automating versioning in every stage
- Documentation in all stages is mostly not automated or developed

B. RQ_2: Development effort

This section focuses on the perceived effort required to develop each stage and the respective tasks as illustrated in Figure 2. To assess the **development effort per stage**, participants were asked to allocate 100 points to each stage based on their development time. According to the median, `Data Handling` requires the most development effort, with 35% of the total effort invested. The dispersion of this stage ranges from 20% to 50%, and 60% of the developers put it within the most effort-intensive stage. The following two stages (`Model Learning` and `Software Development`) each require 20% of the development effort. `System Operations` is the stage with the lowest effort intensity, with a median of 15%.

The following paragraph describes the perceived **effort per tasks**. During the `Data Handling` stage, the participants mostly perceive data cleaning as a high-effort (42%) to medium-effort (32%) task. Over half of the participants perceive data versioning as a low-effort task.

In the `Model Learning` stage, the participants perceive model design, model QA, and model improvement as a high to medium effort task. Again, most participants perceive model versioning as a low-effort task.

As for the `Software Development` and `System Operations` stages, the participants agree that system-level versioning is a low-effort task. In contrast, all the other tasks in both stages require medium effort according to 30-36% of the participants.

- The most development effort intensive stage is `Data Handling`
- High to medium effort is required for data cleaning, model design, model QA, and model improvement
- Participants perceive versioning as a low-effort task

C. RQ_3: Importance of the tasks

Figure 2 presents the importance of the tasks within each stage. Over 65% of the participants perceive data cleaning, transformation, and QA in the `Data Handling` stage as highly important. Whereas data versioning and documentation are of medium and low importance.

A similar pattern is observed in the `Model Learning` stage, where the first four tasks (design, training, model QA, and model improvement) are of high importance, as stated by more than half of the participants, whereas the majority of the responses on versioning and documentation vary from medium to low importance.

All tasks of the `Software Development` stage were indicated as comparably important, with at least 40% of the participants stating that they have high importance. Nearly half of the participants categorized the documentation in this stage as medium important. We also discovered that years of experience positively correlate with the perceived importance of system-level QA with a correlation coefficient of 0.25.

All respective tasks are considered highly important during the `System Operations` stage. For instance, 58% perceive data, model, and system monitoring as highly important.

- All tasks in the four stages, except for versioning and documentation, are perceived as highly important
- Positive correlation for years of experience and perceived importance of system-level QA

D. RQ_4: Factors linked to the adoption of automation

To identify automation factors, we analyzed the correlation between automation and effort at three levels: tasks, stages,

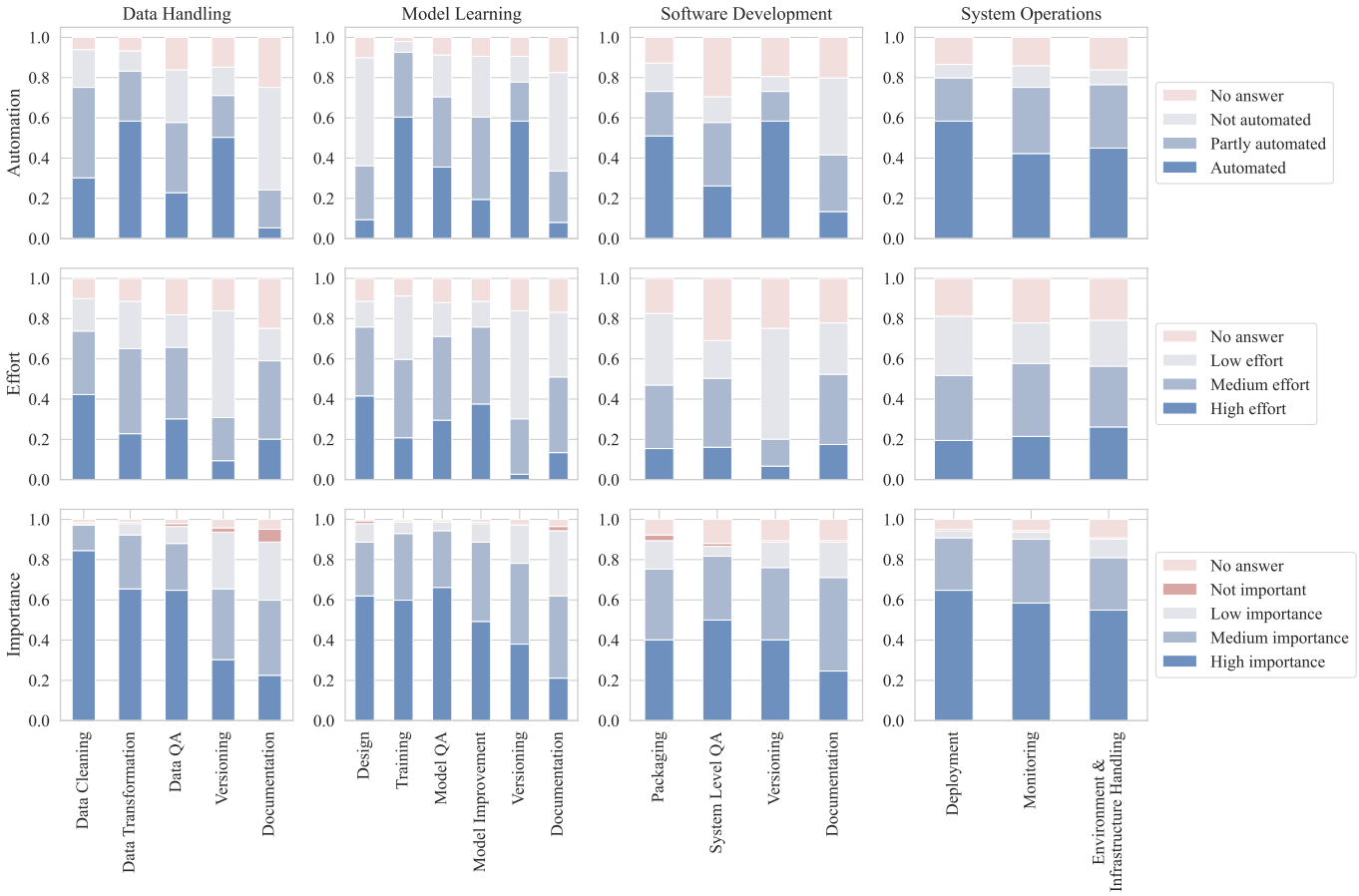

Fig. 2. Automation, development effort, and importance per task

and overall pipeline (Figure 3). We further explored the relation between automation and importance as well as influencing demographic information.

Automation and effort correlation analysis showed a significant negative relationship for most tasks, stages, and the overall pipeline. The detailed results are shown in Figure 3. The effects are significant at level 0.001 or better, varying from -0.29 for data transformation to -0.52 for packaging, which suggests a medium to large effect size. The strongest correlation was observed for the task packaging (-0.52), data QA (-0.47), and software versioning (-0.36). Among stages, the strongest negative correlation appears at the `Software Development` stage (-0.46). Overall pipeline development shows a medium negative correlation (-0.37). We discovered that the healthcare domain has a high negative correlation between automation and effort within all stages (-0.59), especially in the stages `Software Development` (-0.77) and `Model Learning` (-0.59) with a significance level at 0.001.

Automation and importance did not show a significant correlation except a medium positive correlation in the development of the monitoring task (0.29, significant at 0.01) and a slight positive correlation for the `System Operations` stage (0.22). Nevertheless, there is strong evidence that there is little to no relationship between the importance and automation for the other stages and when developing the overall pipeline with correlation coefficients of approximately 0.1.

Other factors - To identify if automation levels differ in different domains, we investigated the dependencies in the most represented domains: finance, e-commerce, and healthcare. The analysis showed that in e-commerce, model training and monitoring are slightly more automated than in other domains (correlations 0.2 and 0.3, respectively). Model improvement is more automated in healthcare (0.2).

When automation is considered in different models, we found an increase in automation by the developers who use regression models (0.3), whereas developers who chose ANN automate data documentation slightly more often (0.2).

In addition, we discovered several relations for tasks automation in different tools: Kuberflow users claimed higher model documentation (0.27), developers using AWS SageMaker indicated higher data cleaning (0.25), data QA (0.26), and model improvement (0.25), developers using Apache Airflow showed higher data QA automation (0.26).

No significant differences in automation were discovered based on team and organization sizes, except for the increase in software versioning and documentation automation in the larger development teams (0.3).

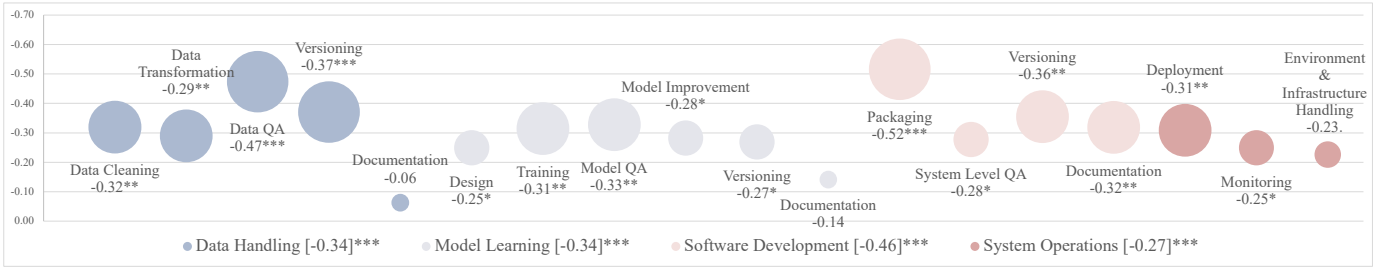

Fig. 3. Correlation analysis of automation and effort for every task. The y-axis and the numbers in the bubble labels show Spearman's correlation coefficients; the bubbles' size represents the significance level. The significance is also indicated next to the correlation coefficients: "." indicates significance at 0.05 level or better. * - 0.01, ** - 0.001, *** indicates significance at 0.0001 level or better.

- Negative correlation between automation and effort
- No correlation between automation and importance

E. RQ_5: Computing resource-intensive tasks

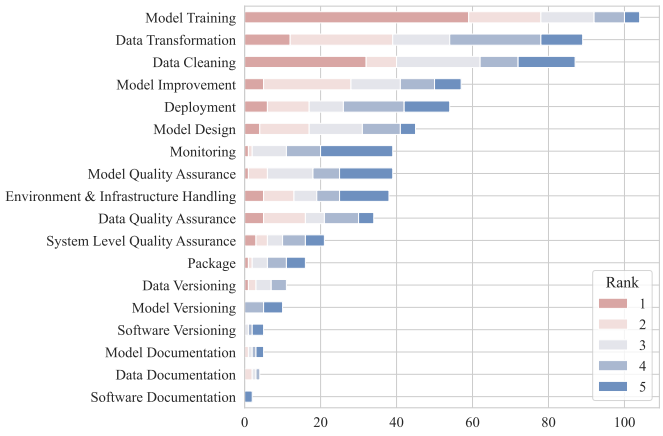

Fig. 4. The ranking of the tasks based on their computing resource intensity

When automating tasks, the majority of participants (93%) take into account the availability of computing resources. Figure 4 shows the tasks ordered by perceived computing resource intensity and their ranking by the participants. Model training, data cleaning, and data transformation were the most resource-intensive tasks. Overall, four out of six tasks from the stage `Model Learning` and three tasks each from the stages `Data Handling` and `System Operations` were placed in the top 10 most resource-intensive tasks. Versioning and documentation-related tasks were considered minor resource-intensive tasks across all stages.

- 93% of participants consider the availability of computing resources
- The most computing resource-intensive tasks according to the practitioners: model training, data transformation, data cleaning, model improvement, and deployment

V. Discussion and Future Work

In this section, we interpret the evidence-based results and link them to relevant literature while providing implications for further research. In addition, we discuss future work and threats to validity towards the end of this section.

Our key findings indicate that tasks in the stage `System Operations` are mostly automated, such as monitoring, as emphasized by Baier et al. [3]. One potential reason may be that automation requires repetitive, deterministic, and standardized tasks where limited creativity is involved [10]. This explains why versioning is mostly automated since practitioners require low effort to develop this task.

This study, however, also identifies that several tasks are mostly partially automated, such as data cleaning. Data is often context-dependent from heterogeneous data sources, requiring individual and domain-specific implementations resulting in a high development effort [3], [19], [22]. Further investigation is crucial to comprehend the reasons and characteristics of partial automation. Once we understand why full automation is impossible, we can develop universal solutions to minimize the development effort in the most development effort-intensive stage `Data Handling`.

Model design and model improvement are unique, high-effort, and important tasks where complete automation is hard to achieve as well. For instance, available data, selected features, suitable algorithms, and hyperparameter optimization highly influence the model design and improvement tasks. Future research can leverage AutoML for identifying optimal settings without human intervention [11].

Moreover, data QA and system level QA require high to medium effort and are mostly partially automated because, according to [12], [22], the benefits of automation do not exceed the effort required for fully automating it. This implies that these tasks require high expertise to develop individual solutions. Notwithstanding, the importance of QA, more precisely system level QA, increases with participant years of experience. We assume that this happens because experienced AI, data, and MLOps engineers might have a deeper understanding of software engineering tasks, the complexity of software development, and the risks associated with inadequate QA [8], [22]. Thus, automated and standardized data QA continuously ensures that the input data is not flawed and bugs cannot be propagated down to the model training to obtain an accurate and reliable AI model [18]. The automation of QA may shift to the highly automated stage `System Operations`, where runtime verification methods assess model performance and bias using production data and information from automated

data, model, and system monitoring tools.

High-effort tasks are less often automated than low-effort tasks, as revealed by this survey's negative correlation between automation and development effort. Certain fields show greater levels of automation. For instance, in the healthcare domain, standardization and automation processes are more prevalent due to stringent regulatory requirements. However, further observations are needed for a comprehensive analysis of the reasons and implications behind this correlation.

In resource-constrained situations, prioritizing the development or improvement of automation for tasks should be based on their importance to companies seeking the benefits of automation. Thus, we suggest focusing on data cleaning, transformation, and QA, model design, training, QA, and deployment first. However, over 20% of the participants did not generate documentation throughout the stages. Thus, participants might not have identified its importance. Documentation is essential to provide accountability, especially for AI models [16]. To ease the process, we imply conducting research on automated solutions that enhance and streamline the documentation process. For instance, Baier et al. [3] and Paleyes et al. [20] recommended documenting concrete actions applied to the data, the model's purpose, technical decisions, and crucial information regarding the software release.

In summary, employing fully automated stages and tasks for the continuous development of AI minimizes the effort during task execution, accelerates model deployment in production, and ensures consistency while improving reliability and quality [14], [20]. Thus, the key implication is to reduce obstacles in fully automating tasks that demand high development effort and significant expertise while prioritizing the most important tasks.

Once the pipeline is better automated, the stated most computing resource-intensive tasks, such as model training, data transformation, data cleaning, model improvement, and deployment, future research can focus on computing resource optimization. For instance, resource consumption can be predicted from previous executions, resources can be allocated more efficiently, and resource-efficient scheduling may be applied [2]. This becomes essential when AI is trained and deployed on edge devices [23].

A. Future Work

This survey explicitly explores the degree of automation and its influencing factors, such as the development effort, importance, and demographic information. Further research may identify other influencing factors via qualitative studies (interviews, literature studies). This paper presents a few domain-specific preliminary findings, but a bigger sample size with sufficient participants within each demographic category would provide deeper insights and further interpretation.

Furthermore, because 93% indicated that they consider computing resources, further research could provide quantitative and more specific insights into these resources for each task, such as CPU or memory. This may help improve resource allocations and the scheduling of automated tasks. Therefore, a case study could measure the computing resources required for available pipelines for the AI development and verify the ranking of this survey.

B. Threats to Validity

This section discusses the four possible threats to validity according to Runeson et al. [25] and how we mitigated them. To improve **internal validity**, we included responses from participants with at least one year of experience. We excluded incomplete surveys to avoid attrition bias. However, we were not able to identify the lack of correlation and low correlation due to the insufficient number of observations. To validate weak correlation associations at a significant level, a larger sample size is required. To enhance **external validity**, we included a diverse group of AI, data, and MLOps engineers from various domains, using different data types, AI models, and tools. We also varied company and team sizes and experience to represent the population better. Threats regarding **construct validity** may occur due to a lack of standardized language and terminologies. To counteract this threat, we based the selected stages and tasks on a literature review, provided clear descriptions, and conducted a pilot study for feedback on wording and response options. However, it is important to recognize that ambiguity may persist regarding participants' understanding of task importance. It remains unclear whether their responses reflect perspectives from the end-users, developers, or task execution within the pipeline. Moreover, we used ordinal scales because we wanted to include "no-answer" options for participants who did not develop a specific task or "not important" if they developed a task but found it unimportant. However, it must be considered that the used scales lack fixed intervals between answer options, which may make it difficult for participants to determine the implied degree of difference between answer choices. Regarding the **reliability**, we published the analysis conducted by two researchers [28]. In addition, we only presented correlations when the significance level was at 1% or better to avoid insignificant linear relationships.

VI. Conclusion

To effectively manage the growing complexity of system-level operations and efficiently integrate rapidly evolving AI models into production systems, automated stages and tasks must be employed to ensure continuous development and monitoring of AI models. These stages include `Data Handling`, `Model Learning`, `Software Development`, and `System Operations`, with their related sets of tasks. This paper aims to gain descriptive insights into practitioners' perspectives on the degree of automation, development effort, importance, and computing resource consumption associated with these stages and tasks. We also examined whether perceived development effort, importance, or demographic information affects automation. Therefore, we collected and analyzed 150 responses through a questionnaire. The key findings indicate that tasks with high effort are automated less frequently than low-effort tasks (e.g., the high-effort tasks, data cleaning, model design, and model improvement, are partially or not automated). Tasks in the stage

`System Operations` are primarily automated, whereas documentation is rarely automated or developed. Perceived importance does not influence the degree of automation (e.g., data cleaning, despite its importance, remains mostly partially automated). 93% of participants consider the availability of computing resources and perceive model training, data cleaning, and data transformation as the most resource-intensive tasks. The discussed key implication is to facilitate automation for development effort-intensive tasks while prioritizing essential tasks to accelerate model deployment and ensure quality.

REFERENCES

[1] Alves, I., Email, U.S.P., Leite, L., Email, S.: Practices for Managing Machine Learning Products : a Multivocal Literature Review Practices for Managing Machine Learning Products : a Multivocal Literature Review pp. 0–23 (2023). https://doi.org/10.36227/techrxiv.21960170.v2

[2] Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., Zimmermann, T.: Software Engineering for Machine Learning: A Case Study. In: 2019 IEEE/ACM 41st International Conference on Software Engineering: Software engineering in practice. pp. 291–300. IEEE, Piscataway, NJ (2019). https://doi.org/10.1109/ICSE-SEIP.2019.00042

[3] Baier, L., Seebacher, S., paper Baier, R.: Challenges in the Deployment and Operation of Machine Learning in Practice (2019), https://www.researchgate.net/publication/332996647

[4] Boucher, P.: Artificial intelligence: How does it work, why does it matter,and what can we do about it? European Parliament, Brussels (2020)

[5] Calefato, F., Lanubile, F., Quaranta, L.: A Preliminary Investigation of MLOps Practices in GitHub. International Symposium on Empirical Software Engineering and Measurement pp. 283–288 (sep 2022). https://doi.org/10.1145/3544902.3546636

[6] Fischer, L., Ehrlinger, L., Geist, V., Ramler, R., Sobiezky, F., Zellinger, W., Brunner, D., Kumar, M., Moser, B.: Ai system engineering—key challenges and lessons learned. Machine Learning and Knowledge Extraction **3**(1), 56–83 (2020)

[7] Fursin, G., Guillou, H., Essayan, N.: CodeReef: an open platform for portable MLOps, reusable automation actions and reproducible benchmarking, http://arxiv.org/pdf/2001.07935v2

[8] Golendukhina, V., Lenarduzzi, V., Felderer, M.: What is software quality for ai engineers? towards a thinning of the fog. Proceedings - 1st International Conference on AI Engineering - Software Engineering for AI, CAIN 2022 pp. 1–9 (2022). https://doi.org/10.1145/3522664.3528599, https://dl.acm.org/doi/10.1145/3522664.3528599

[9] Granlund, T., Kopponen, A., Stirbu, V., Myllyaho, L., Mikkonen, T.: MLOps Challenges in Multi-Organization Setup: Experiences from Two Real-World Cases, http://arxiv.org/pdf/2103.08937v1

[10] Hancock, P.A.: Imposing limits on autonomous systems. Ergonomics **60**, 284–291 (2017). https://doi.org/10.1080/00140139.2016.1190035, http://dx.

[11] He, X., Zhao, K., Chu, X.: AutoML: A survey of the state-of-the-art. Knowledge-Based Systems **212**, 106622 (jan 2021). https://doi.org/10.1016/j.knosys.2020.106622

[12] Hummer, W., Muthusamy, V., Rausch, T., Dube, P., El Maghraoui, K., Murthi, A., Oum, P.: ModelOps: Cloud-based lifecycle management for reliable and trusted AI. In: Proceedings - 2019 IEEE International Conference on Cloud Engineering, IC2E 2019. pp. 113–120. Institute of Electrical and Electronics Engineers Inc. (jun 2019). https://doi.org/10.1109/IC2E.2019.00025

[13] John, M.M., Holmström Olsson, H., Bosch, J.: Architecting AI Deployment: A Systematic Review of State-of-the-Art and State-of-Practice Literature. Lecture Notes in Business Information Processing **407**, 14–29 (2021). https://doi.org/10.1007/978-3-030-67292-8_2/COVER, https://link.springer.com/chapter/10.1007/978-3-030-67292-8_2

[14] John, M.M., Olsson, H.H., Bosch, J.: Towards MLOps: A Framework and Maturity Model. Proceedings - 2021 47th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2021 pp. 334–341 (sep 2021). https://doi.org/10.1109/SEAA53835.2021.00050

[15] Karlaš, B., Interlandi, M., Renggli, C., Wu, W., Zhang, C., Mukunthu Iyappan Babu, D., Edwards, J., Lauren, C., Xu, A., Weimer, M.: Building Continuous Integration Services for Machine Learning. In: Gupta, R. (ed.) Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2407–2415. ACM Digital Library, Association for Computing Machinery, New York,NY,United States (2020). https://doi.org/10.1145/3394486.3403290

[16] Königstorfer, F., Thalmann, S.: AI Documentation: A path to accountability. Journal of Responsible Technology **11**, 100043 (oct 2022). https://doi.org/10.1016/J.JRT.2022.100043

[17] Kreuzberger, D., Kühl, N., Hirschl, S.: Machine Learning Operations (MLOps): Overview, Definition, and Architecture. IEEE Access **11**, 31866–31879 (2023). https://doi.org/10.1109/ACCESS.2023.3262138, https://ieeexplore.ieee.org/document/10081336/

[18] Lwakatare, L.E., Crnkovic, I., Rånge, E., Bosch, J.: From a Data Science Driven Process to a Continuous Delivery Process for Machine Learning Systems. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **12562 LNCS**, 185–201 (2020). https://doi.org/10.1007/978-3-030-64148-1_12/TABLES/3, https://link.springer.com/chapter/10.1007/978-3-030-64148-1_12

[19] Makinen, S., Skogstrom, H., Laaksonen, E., Mikkonen, T.: Who needs MLOps: What data scientists seek to accomplish and how can MLOps help? Proceedings - 2021 IEEE/ACM 1st Workshop on AI Engineering - Software Engineering for AI, WAIN 2021 pp. 109–112 (may 2021). https://doi.org/10.1109/WAIN52551.2021.00024

[20] Paleyes, A., Urma, R.G., Lawrence, N.D.: Challenges in Deploying Machine Learning: A Survey of Case Studies. ACM Computing Surveys **55**(6) (dec 2022). https://doi.org/10.1145/3533378, https://doi.org/10.1145/3533378

[21] Rahman, A., Bhuiyan, F.A., Hassan, M.M., Shahriar, H., Wu, F.: Towards Automation for MLOps: An Exploratory Study of Bot Usage in Deep Learning Libraries. Proceedings - 2022 IEEE 46th Annual Computers, Software, and Applications Conference, COMPSAC 2022 pp. 1093–1097 (2022). https://doi.org/10.1109/COMPSAC54236.2022.00171

[22] Raj, A., Bosch, J., Olsson, H.H., Wang, T.J.: Modelling data pipelines. In: 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). pp. 13–20. IEEE (26-28082020). https://doi.org/10.1109/SEAA51224.2020.00014

[23] Raj, E., Buffoni, D., Westerlund, M., Ahola, K.: Edge MLOps: An Automation Framework for AIoT Applications. Proceedings - 2021 IEEE International Conference on Cloud Engineering, IC2E 2021 pp. 191–200 (2021). https://doi.org/10.1109/IC2E52221.2021.00034

[24] Recupito, G., Pecorelli, F., Catolino, G., Moreschini, S., Nucci, D.D., Palomba, F., Tamburri, D.A.: A Multivocal Literature Review of MLOps Tools and Features (October) (2022). https://doi.org/10.13140/RG.2.2.10257.71526/1, https://github.com/gilbertrec/MLR-MLOps-Tools-Features-Appendix

[25] Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering **14**(2), 131–164 (apr 2009). https://doi.org/10.1007/S10664-008-9102-8/TABLES/27, https://link.springer.com/article/10.1007/s10664-008-9102-8

[26] Serban, A., van der Blom, K., Hoos, H., Visser, J.: Adoption and Effects of Software Engineering Best Practices in Machine Learning. In: Proceedings of the 14th ACM. pp. 1–12. ACM Digital Library, Association for Computing Machinery, New York,NY,United States (2020). https://doi.org/10.1145/3382494.3410681

[27] Steidl, M., Felderer, M., Ramler, R.: The pipeline for the continuous development of artificial intelligence models—Current state of research and practice. Journal of Systems and Software **199**, 111615 (may 2023). https://doi.org/10.1016/J.JSS.2023.111615

[28] Steidl, M., Golendukhina, V., Felderer, M., Ramler, R.: Continuous Development of AI: a Practitioners' Survey (apr 2023). https://doi.org/10.5281/ZENODO.7838983, https://zenodo.org/record/7838983

[29] Stone, P., Brooks, R., Brynjolfsson, E., Calo, R., Etzioni, O., Hager, G., Hirschberg, J., Kalyanakrishnan, S., Kamar, E., Kraus, S., Leyton-Brown, K., Parkes, D., Press, W., Saxenian, A., Shah, J., Tambe, M., Teller, A.: Artificial Intelligence and Life in 2030: One Hundred Year Study on Artificial Intelligence: Report of the 2015-2016 Study Panel. Stanford University (2016), https://ai100.stanford.edu/2016-report

[30] Testi, M., Ballabio, M., Frontoni, E., Iannello, G., Moccia, S., Soda, P., Vessio, G.: MLOps: A Taxonomy and a Methodology. IEEE Access **10**, 63606–63618 (2022). https://doi.org/10.1109/ACCESS.2022.3181730

[31] Zhou, Y., Yu, Y., Ding, B.: Towards MLOps: A Case Study of ML Pipeline Platform. Proceedings - 2020 International Conference on Artificial Intelligence and Computer Engineering, ICAICE 2020 pp. 494–500 (oct 2020). https://doi.org/10.1109/ICAICE51518.2020.00102