A BACKGROUND

A.1 CI/CD practices in traditional software engineering and DevOps

Continuous integration (CI) practices within agile development approaches [2] have widespread usage in today's software development community. CI refers to frequent integration of software artifacts (e.g., code base, dependencies) during development [29]. The benefits of CI include reducing release cycles by keeping software always ready for release, increasing team productivity, finding bugs as early as possible, and improving software quality [28, 29]. On the other hand, continuous deployment (CD) indicates the automatic and frequent delivery of software to the customer or production environment [17, 27, 29]. This practice allows for the customer to receive early feedback on the functionality delivered of the software, helping to identify further improvements [5]. CI/CD contributes to reducing risks and improving software quality [9]. DevOps (Dev for development and Ops for operations) is a natural evolution of agile and CI/CD practices. It is a paradigm of continuous and collaborative software engineering [17]. It aims to deliver software more frequently using an automated, repeatable, and continuous process flow. Furthermore, DevOps involves practices, tools, and sociotechnical aspects that facilitate easier collaboration between development and operations teams (PS42).

A.2 MLOps

CI/CD practices have been very useful in the development of ML-enabled systems, much like their success in traditional software development. Based on its benefits in traditional software engineering, the ML development community adopts CI / CD practices in ML development [20]. The experimental and probabilistic nature of AI/ML systems requires frequent repetition of development and deployment flow. Therefore, the ML development and deployment process demands practices similar to those of CI / CD (PS3). MLOps concept has emerged with the goal of establishing a set of practices for more efficient ML development and deployment flow by mimicking the DevOps practices (PS1). MLOps introduces additional practices specific to ML, emphasizing the automation of development and monitoring at all stages of ML, and deployment (PS1), (PS10). One important practice of MLOps is monitoring. Monitoring helps to track data and model performance shifts to ensure the reliability and quality of system over time (PS7), and enables continuous feedback loop between development, operations, and production environments. At this point, along with CI and CD, MLOps incorporates the continuous training (CT) concept, because the continuous delivery of an ML system is not independent from data collection, analysis, and model training phases (PS78). Further, MLOps introduces new practices regarding the roles, responsibilities and collaboration between people with different backgrounds. MLOps engineer is a new role introduced with MLOps, referring to the professionals responsible for maintaining the operational stability of ML pipelines and the entire ML system (PS106). Moreover, new collaboration practices, i.e., collaboration points (PS150), have been introduced to improve the efficiency of MLOps projects by enabling better collaboration between data science and software engineering teams. Here we also mention DataOps and ModelOps which are two concepts that share similarities with MLOps. However, our work focuses on MLOps. Thus, we only highlight differences between concepts and only cover MLOps in our RQs. DataOps is coined to refer to the automation of data analytics life-cycle [11]. DataOps establishes a set of practices for data-centric systems mimicking DevOps practices and aims to ship data frequently in an automated [33]. ModelOps is defined as continuous practices for AI-enabled software systems (PS33). The key activities revolve around continuous model training, model metadata versioning, deployment, and management of AI pipelines. While both DataOps and ModelOps are related to MLOps, these three terms refer to different concepts. DataOps centralizes data analytics, management, and the shipment of data. ModelOps centralizes building and managing

various versions of models and serving them in AI systems. On the other hand, MLOps is primarily concerned with productionalization of ML systems and automating the whole ML pipeline from data collection through model training, deployment, and monitoring. Further, the skills required by these three concepts vary; DataOps mainly involves data science, data analysis, and data engineering, and ModelOps involves data science. MLOps utilizes a broader range of skills such as data and ML engineering, data science, and MLOps engineering (see Section 3.3).