Classification and Challenges of Non-Functional Requirements in ML-Enabled Systems: A Systematic Literature Review

Vincenzo De Martino, Fabio Palomba

Software Engineering (SeSa) Lab - Department of Computer Science, University of Salerno (Italy)

Abstract

Context: Machine learning (ML) is nowadays so pervasive and diffused that virtually no application can avoid its use. Nonetheless, its enormous potential is often tempered by the need to manage non-functional requirements (NFRs) and navigate pressing, contrasting trade-offs.
Objective: In this respect, we notice a lack of systematic synthesis of challenges explicitly tied to achieving and managing NFRs in ML-enabled systems. Such a synthesis may not only provide a comprehensive summary of the state of the art but also drive further research on the analysis, management, and optimization of NFRs of ML-enabled systems. **Method:** In this paper, we propose a systematic literature review targeting two key aspects such as (1) the classification of the NFRs investigated so far, and (2) the challenges associated with achieving and managing NFRs in ML-enabled systems during model development Through the combination of well-established guidelines for conducting systematic literature reviews and additional search criteria, we survey a total amount of 130 research articles.
Results: Our findings report that current research identified 31 different NFRs, which can be grouped into six main classes. We also compiled a catalog of 26 software engineering challenges, emphasizing the need for further research to systematically address, prioritize, and balance NFRs in ML-enabled systems.
Conclusion: We conclude our work by distilling implications and a future outlook on the topic.

Keywords: Software Engineering for Artificial Intelligence; Non-Functional Requirements; Systematic Literature Reviews.

1. Introduction

Machine learning (ML) is now, more than ever, being used in theory, experiment, and simulation [54, 57]. On the one hand, companies and individuals increasingly rely on the outcome of machine learning models to make informed decisions [79] or automate tasks that would take substantial human workload [56]. On the other hand, machine learning-intensive systems, i.e., systems that embed machine learning solutions, have been recently deployed in multiple domains, with some recent applications showing highly efficient and accurate performance [48, 53]. As such, the pervasiveness of machine learning-intensive systems is expected to increase further in the coming years in multiple domains [55, 78].

Email addresses: vdemartino@unisa.it (Vincenzo De Martino), fpalomba@unisa.it (Fabio Palomba)

Nonetheless, such a pervasiveness is constantly threatened by multiple concerns, which are often not related to the specific features made available to users, but to non-functional attributes [26, 51]. In particular, a non-functional requirement (NFR) is defined as a condition that specifies a criterion that may be used to judge the operation of a system rather than specific behaviors [12]. In the context of ML-enabled systems, non-functional attributes may affect the overall level of reliability, trustworthiness, and sustainability of these systems [24, 27, 42]. It is therefore not surprising that the software engineering research community—and specifically the software engineering for artificial intelligence (SE4AI) research branch—has been investing notable efforts in understanding non-functional requirements of ML-enabled systems, other than proposing methods and instruments to support practitioners when dealing with them [9, 23, 30]. This effort is also stimulated by government and funding agencies, which are more and more willing to invest in the matter, e.g., the European Union has recently approved the *Artificial Intelligence Act*,[1] which aims at promoting research on themes connected to the improvement of non-functional attributes of artificial intelligence-based software systems.

Recent advances in the field of SE4AI contributed to the development of a consistent body of knowledge with respect to the management of multiple non-functional requirements, including fairness [13, 19, 76], security [25, 40], privacy [39, 64], and more [35]. While recognizing the relevant advances made over the last years, our research identifies a notable key limitation.

> ⚠ *Despite the current, extensive body of knowledge produced by the SE4AI research community regarding the management of non-functional requirements in ML-enabled systems, there is still a lack of a comprehensive, systematic synthesis of the current knowledge on the* ***non-functional requirements impacting ML-enabled systems*** *and the* ***challenges encountered in addressing them within the context of these systems.***

An improved understanding of these aspects may have critical implications for researchers and practitioners. First, researchers might learn more about the current state of the art, possibly identifying neglected research angles that would be worth further investigating. At the same time, practitioners may have a comprehensive overview of the instruments that researchers have been providing to support the analysis and optimization of non-functional requirements, possibly accelerating the technological transfer of academic prototypes to industry. In this paper, we conduct a systematic literature review (SLR) on non-functional requirements of ML-enabled systems. In particular, we focus on traditional ML and DL systems but do not address large language models (LLM) or generative artificial intelligence systems integration. By focusing on traditional ML and DL systems, this study aims to provide a comprehensive synthesis of the non-functional requirements related to these technologies, which remain critical in a wide range of applications. Our work follows well-established guidelines [33, 72] and additional search criteria based on seed set identification [49] to comprehensively synthesize existing research. From an initial set composed of over 3,000 hits, and after applying multiple snowballing rounds and additional data collection procedures, we ended up analyzing more than 94,000 research results. Through the application of exclusion/inclusion criteria and quality assessment, we finally selected 130 papers. Additionally, we provide a new catalog consisting of 26 software engineering challenges proposed by researchers to assist practitioners in optimizing ML-enabled systems specifically in relation to non-functional requirements during the software life cycle. These challenges are not

[1] The European Union Artificial Intelligence Act: `https://artificialintelligenceact.eu`.

about the NFRs themselves but instead arise from the complexities faced by ML-enabled systems when striving to meet specific non-functional requirements. They occur at various stages of the software life cycle, including design, development, implementation, and operation, and are shaped by the socio-technical context and the inherent trade-offs between competing NFRs. We conclude the paper by elaborating on the implications of our results, along with the actionable items that readers of our work may (re-)use to analyze further the problem of non-functional requirements in machine learning-intensive software systems.

2. Related Work and Motivation

To the best of our knowledge, no systematic literature review has been conducted to classify the non-functional requirements specific to ML-enabled systems and to summarize the challenges encountered in achieving these requirements during their software life cycle. At the same time, it is important to point out that some secondary studies recently attempted to (1) synthesize the research on software engineering for artificial intelligence [43], (2) explore, in a preliminary fashion, the relevance and research interest around non-functional requirements of ML-enabled systems [23], and (3) summarize some of the key academic and industrial challenges faced by practitioners in managing functional and non-functional requirements in ML-enabled systems [2, 24, 27, 67]. These works are closely related to ours, but they either lack the comprehensive scope or the detailed synthesis of challenges specifically tied to NFRs in ML-enabled systems.

First and foremost, Martinez-Fernandez et al. [43] conducted a systematic mapping study of the research on software engineering for artificial intelligence. The main goal of the work was to provide a comprehensive schema representing the elements composing the field of SE4AI, from requirements engineering to verification and validation. In other terms, the systematic mapping study had a pretty broad objective and aimed at covering all the research on the matter. As such, there are multiple differences in our study. While our scope is limited to non-functional requirements, we aim to address the matter in a more detailed fashion by letting emerge a complete set of non-functional requirements discussed in the literature, other than the challenges of ML-enabled systems associated with their management. Secondly, ours is a systematic literature review rather than a mapping study: as such, there are intrinsic, methodical differences in the search process conducted and in the criteria used to select the relevant pieces of research. Third, we additionally tuned the search process to produce an extensive set of seed papers, as further discussed in Section 3—hence attempting to strengthen the completeness of the search process.

Habibullah et al. [23] recently investigated the topic of non-functional requirements of ML-enabled systems under three perspectives such as (1) the clustering of non-functional attributes based on shared characteristics; (2) the estimation of the number of relevant studies that investigated aspects connected to non-functional attributes; and (3) the definition of the scope of non-functional requirements. Habibullah et al. [23] shared the same overall objective of our paper, i.e., an improved understanding of non-functional requirements of ML-enabled systems. Nonetheless, we aimed to conduct a comprehensive, systematic literature review. At the same time, we broadened our scope to encompass a wider range of research perspectives, including an in-depth investigation of the challenges for ML-enabled systems specifically relating to non-functional requirements.

Horkoff [27] discussed the challenges that the requirements engineering research community would be called to face to address themes connected to non-functional requirements of ML-enabled systems. In this position paper, the author identified several challenges based on the extensive experience accumulated in the industry over the years. Our systematic literature review

aims to collect comprehensive pieces of information coming from the scientific community to provide insights into how researchers have defined and addressed the problem so far.

Habibullah and Horkoff [24] conducted an interview-based study with ten machine learning engineers to elicit the practices used to face non-functional requirements. In particular, the authors were interested in collecting information about the identification and measurement mechanisms put in place, the importance of various non-functional requirements from their perspective, and the challenges associated with the identified non-functional requirements. This work can be, therefore, seen as complementary to our systematic literature review. We enlarge the knowledge of non-functional requirements in ML-enabled systems by synthesizing the current literature from various perspectives. Additionally, our work includes and further extends the findings on the challenges identified through a different research method.

Villamizar et al. [67] explored multiple aspects of how the Requirements Engineering (RE) community has contributed to advancing the knowledge on requirements engineering practices in ML-enabled systems, including the type of RE contributions and topics, the quality characteristics considered, the empirical evaluations conducted, and the future directions on the interplay between RE and ML. The article was published in 2021 and covered the research articles published by the end of 2020. The authors identified 35 primary studies by executing a search query that looked for articles containing terms such as "software" (or synonyms like "applications" and "systems"), "machine learning", and "requirements engineering". We identified several differences when comparing this work to ours. In the first place, the scope and breadth of the analysis. Villamizar et al. [67] aimed to map the existing body of knowledge on RE research and provide a broad overview of the contributions in the field. In doing so, the authors focused on any kind of research activity, looking at both functional and non-functional requirements. On the contrary, our work is more specific, i.e., targets non-functional requirements only and does not aim at mapping the existing knowledge but rather classifies non-functional requirements and synthesizes the challenges to deal with them. The methodology employed in our work, i.e., a systematic literature review, as opposed to a mapping study procedure used by Villamizar et al. [67], allows for a more in-depth and structured analysis of the identified studies, enabling us to provide a more comprehensive understanding of non-functional requirements in ML-enabled systems. Perhaps more importantly, Villamizar et al. [67] limited the search to the papers that explicitly used terms like "machine learning" and "requirements engineering". On the one hand, our search string enlarges the breadth of the analysis, as it considers articles focusing on additional technologies such as artificial intelligence, deep learning, reinforcement learning, and deep neural networks, hence obtaining a higher coverage of the theme. On the other hand, the hybrid nature of our systematic article collection procedure allowed the extraction of primary studies that addressed non-functional requirements even without an explicit reference to "requirements engineering". As further explained in Section 3, this represents a key point to ensure completeness. Indeed, several primary studies addressed specific non-functional requirements, e.g., fairness, without any explicit mention of terms like "requirements engineering" and "non-functional requirements". As a consequence, our hybrid systematic literature review is, by design, larger in terms of coverage. As proof of that, the amount of primary studies included in our analysis is significantly larger: while Villamizar et al. [67] extracted 35 primary studies, we reached 130 papers. Considering our focus on non-functional requirements, we regarded the coverage obtained as comprehensive and thorough, reflecting a broader and more detailed examination of the challenges and considerations related to non-functional requirements in ML systems. In addition to the considerations above, Villamizar et al. [67] identified three main challenges for the entire RE community related to (1) lack of knowledge regarding non-functional requirements, (2)

lack of validation techniques, and (3) handling customers' expectations. Our work addresses one of these key challenges, i.e., the lack of knowledge regarding non-functional requirements, by systematically surveying the existing literature on the matter and summarizing the more specific challenges to dealing with non-functional requirements. For this reason, our work should also be considered as a complementary valuable source that focuses on a key matter for RE research.

Ahmad et al. [2] reported on an interview-based study aimed at eliciting the existing industrial guidelines and best practices. The authors involved 29 practitioners and discussed the integration of human-centered aspects in requirements engineering for AI, emphasizing the gap between literature and practice through practitioner interviews. Our work is clearly complementary. First, we focused on the work done by researchers by performing a systematic literature review, as opposed to survey practitioners. Second, we aimed at classifying non-functional requirements and associated challenges, as opposed to investigating the current practices and best practices. As such, the paper by Ahmad et al. [2] can be seen as orthogonal to ours.

Ahmad et al. [1] also performed a systematic mapping study on RE4AI systems. Their work focused on four key objectives: (1) identifying RE frameworks, notations, modeling languages, and tools; (2) surveying the evaluation methods used to assess the retrieved available methods; (3) informing the community about the target application domains for which the available methods are available; and (4) identifying the limitations and challenges of the methods. Also in this case, our work can be seen as complementary. First, Ahmad et al. [1] focused specifically on automated methods to handle the whole requirements engineering process, as opposed to our focus on classifying non-functional requirements and identifying associated challenges. Second, the challenges identified by Ahmad et al. [1] mainly pertained to the automated methods to handle functional requirements: in this sense, our work extends the currently available knowledge on the challenges faced by ML-enabled systems in managing non-functional requirements.

Other recent papers analyzed aspects that are tangentially connected to the themes of our work. The article by Damirchi and Amineh [15] presented a non-systematic survey focusing on a specific problem of non-functional requirements engineering, namely the elicitation procedures. The paper by Cheverda et al. [14] proposed a preliminary work aiming at creating a taxonomy of properties to design intelligent systems, while the paper by Ali et al. surveyed the literature on quality models for AI systems [4]. The paper by Ronanki et al. [59] focused on the ethical requirements of engineering processes. The paper by Krishna et al. [59] focused on software quality models for AI systems. Our paper is complementary to these articles, both considering its larger scope and its systematic connotation.

While the papers discussed above are closely connected to the work proposed herein, it is also worth mentioning the existence of an ever-increasing number of secondary studies targeting multiple aspects of software engineering for artificial intelligence research.

A consistent amount of systematic literature reviews focused on the synergies between artificial intelligence and software engineering [69], other than on the software engineering challenges and solutions for developing artificial intelligence systems [21, 35, 42, 50]. Our work focuses on non-functional requirements, hence providing finer-grained pieces of information and insights on the next steps that researchers should consider to better support practitioners. Other researchers targeted the quality assurance problem, which is deemed one of the most relevant and complex for the SE4AI research community. In particular, we identified systematic literature review and empirical investigations into the field of software quality [20, 46], design patterns [60, 71], software architecture [61], and testing [10, 11, 58, 77]. Finally, recent systematic literature reviews have been conducted with the aim of understanding the deployment strategies for artificial intelligence systems [29], and model-based development approaches [41].

Based on the consistent body of knowledge summarized above, we can draw two main conclusions. First, the attention given to requirements engineering practices highlights the relevance of the problem and the importance of understanding the aspects that characterize ML-enabled systems and the challenges that need further exploration. Second, previous work, such as that by Villamizar et al. [67], emphasizes the need to closely investigate non-functional requirements and expand our knowledge in this area. Our systematic literature review addresses existing knowledge gaps by classifying non-functional requirements of ML-enabled systems and presenting a *superset* of the challenges identified in previous research.

3. Research Method

The *goal* of the study was to survey the current research on non-functional requirements of machine learning-enabled systems, with the *purpose* of providing researchers with actionable items and insights that they can exploit to further improve the support provided to machine learning engineers in the *context* of machine learning and deep learning-enabled systems. The *perspective* is of the broad community of researchers working on software engineering for artificial intelligence (SE4AI). At first, fresh Ph.D. students may be particularly interested in our results since they may be looking for neglected or emerging research areas to investigate in the context of their doctoral studies. Senior researchers may be looking for research opportunities to advance the current state of the art or additional opportunities for research proposals and industrial collaborations. By senior researchers, we mean all academics who may be interested in contributing to advancing the current knowledge on non-functional requirements of ML-enabled systems, hence extending to anybody who may help advance the state of the art in the field. Finally, practitioners - which we identify as professional software engineers, machine learning engineers, project managers, and tool vendors - may be interested in understanding the challenges to face to deal with non-functional requirements, as these may possibly raise obstacles to technological transfer that should be carefully considered in practice.

We designed a systematic literature review (SLR) following the guidelines by Kitchenham et al. [32]. The initial use of a search string focusing on terms such as "NFR" or "non-functional requirement" aimed to let the types of NFRs emerge naturally from the data, indeed following a bottom-up approach. To mitigate threats due to incompleteness of the search, we boosted the search process by means of two additional steps. These steps incorporated a top-down approach, which leveraged implicit understanding of NFRs to guide the identification and selection of relevant studies. This approach ensured the use of broader conceptual insights to complement the systematic search process, addressing potential gaps and improving the overall comprehensiveness of the review. First, we systematically screened the research articles published in top-tier software engineering and artificial intelligence venues to identify seed papers to further process [74]. Second, we integrated the snowballing procedure [73], i.e., the iterative scanning of the incoming and outcoming references of the primary studies done to identify additional relevant sources of information. Our systematic literature review can be considered *"hybrid"* [49], as it combines traditional search strategies with additional search steps and snowballing. In terms of reporting, we followed the *ACM/SIGSOFT Empirical Standards*[2] and, in particular, the *"General Standard"* and *"Systematic Reviews"* guidelines.

[2]The *ACM/SIGSOFT Empirical Standards*: `https://github.com/acmsigsoft/EmpiricalStandards`.

3.1. Research Objectives and Questions

We defined two research questions (**RQ**s): $\mathbf{RQ}_1$ aimed at systematically classifying the non-functional requirements identified by researchers, as well as providing insights into the application domains considered by researchers. By 'application domain', we referred to the specific application areas or contexts in which a non-functional requirement has been investigated, such as healthcare, finance, automotive, and others. Additionally, a domain can also be general, encompassing non-functional requirements that are broadly applicable across any kind of domain without being tied to a specific context. More specifically, we asked:

> **RQ$_1$:** *What are the non-functional requirements of ML-enabled software considered by researchers and in which domains were they addressed?*

From a *scientific* perspective, by addressing this research question we aimed to contribute to classifying the non-functional requirements of ML-enabled systems [23]. While previous efforts in this respect have been conducted, there is still not a systematic classification - this represents a key contribution to our work. From a *practical* perspective, the classification of non-functional requirements serves three key purposes. In the first place, a clear classification allows to aggregate the current knowledge on the matter, enabling more targeted research and development efforts. Researchers can indeed use the classification to identify specific areas that need further investigation, ensuring that their efforts are focused and aligned with industry needs. Second, the classification provides an instrument to increase the community awareness of the multifaceted aspects affecting the design of ML-enabled systems: researchers and practitioners may use the classification to have a comprehensive overview of the non-functional requirements that should be preserved, potentially identifying areas where trade-offs might be necessary. Finally, as for practitioners, understanding the various non-functional requirements and their classifications aids in the design and implementation of ML-enabled systems. It ensures that all relevant aspects are considered, leading to the development of more robust, reliable, and efficient systems.

$\mathbf{RQ}_2$ aimed to aggregate and synthesize the body of knowledge on the challenges faced by ML-enabled systems when addressing non-functional requirements, informing researchers and practitioners on the aspects to further consider while developing these systems.

> **RQ$_2$:** *What are the challenges for ML-enabled systems specifically relating to non-functional requirements?*

From a *scientific* perspective, addressing this research question advances the body of knowledge by systematically aggregating the challenges for ML-enabled systems specifically relating to non-functional requirements, hence proposing a super-set of the challenges identified in previous work. This classification provides a foundational understanding that can guide future research efforts, ensuring that subsequent studies build upon a well-defined framework. It also highlights gaps in the existing literature, pointing researchers towards areas that require further investigation and potentially leading to the development of new theories, models, and methodologies specific to ML-enabled systems. From a *practical* perspective, understanding the challenges for ML-enabled systems specifically relating to non-functional requirements equips practitioners with crucial insights that can enhance the design, development, and maintenance of these systems. By identifying common challenges, practitioners can develop strategies and best practices

to mitigate them, leading to more robust, reliable, and efficient ML-enabled software. Additionally, this knowledge helps in setting realistic expectations and priorities during the software development lifecycle, ensuring that critical non-functional requirements are adequately addressed, thereby improving overall system quality and user satisfaction.

3.2. Research Method to Conduct the Systematic Literature Search

We applied the guidelines by Kitchenham et al. [33] to identify primary studies.

3.2.1. Research Query definition

We first extracted relevant terms from the research questions, identifying the keywords from the **RQs** [33]. We then elaborated on the alternative spellings and synonyms for all the terms. Afterward, we used boolean operators to assemble the search string, i.e., we used the *'OR'* operator for the concatenation of alternative spellings and synonyms, while the *'AND'* operator for the concatenation of relevant terms. We elaborated on the following search string:

Search String.

(("Machine Learning") OR ("Artificial Intelligence") OR ("Deep Learning") OR ("Reinforcement Learning") OR ("Deep Neural Network") OR ("ml") OR ("ai") OR ("dl")) AND (("nfr") OR ("Non-Functional Requirement") OR ("Non Functional Requirement*"))*

The search string included terms connected to machine, deep, and reinforcement learning, but also to deep neural networks and artificial intelligence. This was done to deal with the lack of a standard terminology: it is indeed possible that researchers used more generic terms, like *"Artificial Intelligence"*, or more specific terms, like *"Deep Neural Networks"* to indicate the analysis of ML-enabled systems.

Secondly, the search string did not include terms related to any specific, known non-functional requirements, e.g., fairness, but focused on the more generic concept of non-functional requirement, including keywords like *"Non-Functional Requirement"*, *"Non Functional Requirement"*, and *"NFR"*. We are aware that a search string including the specific non-functional requirements would have ensured the collection of a larger amount of relevant papers. Nonetheless, we would have faced two issues. First, we would have been bound to include the non-functional requirements explicitly classified in previous studies while defining the search string. As such, we could not have satisfactorily addressed $\mathbf{RQ}_1$, unable to classify additional non-functional requirements emerging from primary studies. Second, the systematic literature review would have become prohibitive in terms of effort. For instance, Habibullah et al. [24] estimated the number of hits for a search query that includes all classified non-functional requirements in over 200,000 articles: such a search query would have had a low precision, identifying a high amount of irrelevant resources; also, the application of exclusion and inclusion criteria over such a large set of candidate articles would have required an excessive effort. Hence, we identified an alternative solution, opting to implement a hybrid mechanism to search additional relevant resources.

Considering the points discussed, we opted for a broader search string to enhance the search's recall, aiming to gather more papers while maintaining the sustainability of our data collection and analysis. This choice inevitably affected the search's precision and increased the effort needed for applying exclusion and inclusion criteria. However, we embraced this trade-off to ensure the inclusion of all pertinent sources in our study.

3.2.2. Search Databases

We applied the search on *ACM Digital Library*,[3] *Scopus*,[4] and *IEEEXplore*.[5] These digital libraries are often used to carry out systematic literature and mapping studies, providing a comprehensive set of resources to conduct them. It is worth noticing that *Scopus* indexes all the papers published by relevant publishers such as *Springer* and *Elsevier* - this is the reason why we did not include the *SpringerLink*[6] and *ScienceDirect*[7] databases. At the same time, we still opted for the inclusion of *ACM Digital Library* and *IEEEXplore*. This was done because the proceedings of some relevant ACM and IEEE conferences might not have been indexed by *Scopus* and, for this reason, we might have missed relevant resources for our study.

3.2.3. Exclusion and Inclusion criteria

The papers retrieved from the search process were assessed against the following exclusion and inclusion criteria [32].

Exclusion criteria. The resources that met the constraints reported below were excluded:

- **EC1**: Papers not written in English;
- **EC2**: Duplicated papers;
- **EC3**: Papers whose full-text read was not available;
- **EC4**: Paper not published or not peer-reviewed;
- **EC5**: Workshop, vision, systematic, survey papers;
- **EC6**: Short papers, with a page count of less than five pages;
- **EC7**: Papers that did not fall into themes of computer science and computer engineering;
- **EC8**: Papers published before 2012;
- **EC9**: Conference papers which were later extended as journal submissions;
- **EC10**: Papers that do not address any non-functional requirement;
- **EC11**: Papers that do not address non-functional requirements in ML-enabled systems, e.g., those discussing non-functional requirements in general software systems without a specific focus on ML components.

The exclusion criteria were designed to filter out papers that would not contribute meaningfully to our study. **EC1** excluded papers not written in English to ensure language consistency and comprehensibility. **EC2** removed duplicate papers to avoid redundancy. Papers whose full-text read was not available (**EC3**) were excluded to ensure we had complete information for analysis. **EC4** excluded papers not published or not peer-reviewed to maintain the quality and reliability of the sources. **EC5** excluded workshop, systematic, or survey papers to focus on original research contributions. In particular, we excluded the workshops because they often lack the maturity and validation found in full conference or journal papers. **EC6** excluded

[3] *ACM Digital Library*: `https://dl.acm.org`.
[4] *Scopus*: `www.scopus.com`.
[5] *IEEEXplore*: `http://ieeexplore.ieee.org`.
[6] *SpringerLink*: `https://link.springer.com/`.
[7] *ScienceDirect*: `https://www.sciencedirect.com`.

short and vision papers to ensure sufficient content for thorough analysis. **EC7** excluded papers outside the themes of computer science and computer engineering to maintain relevance to our study. **EC8** excluded papers published before 2012 to ensure the relevance and timeliness of the data. **EC9** excluded conference papers later extended as journal submissions to avoid duplicated content. **EC10** and **EC11** ensured that only papers addressing non-functional requirements in the specific context of ML-enabled systems were included, maintaining the focus and relevance of the study.

Inclusion criteria. We included the resources that met at least one of the following constraints:

- **IC1**: Papers must address non-functional requirements in ML-enabled systems. This includes using specific non-functional requirements terminology (e.g., fairness, performance, scalability) within the context of machine learning-enabled systems.
- **IC2**: Papers must focus on specific types of non-functional requirements relevant to ML-enabled systems. This means the paper should primarily revolve around one or more non-functional requirements, rather than just mentioning them.
- **IC3**: Papers must identify and describe specific challenges associated with non-functional requirements in ML-enabled systems. This includes outlining the nature of the challenges, their causes, and potential impacts.
- **IC4**: Papers must explicitly discuss the non-functional requirements of ML-enabled systems, either within a general domain (applicable across multiple contexts) or within a clearly defined specific domain (e.g., healthcare, finance).

The inclusion criteria were structured to exclude papers that defined NFRs or listed challenges without connecting them to ML-enabled systems. This ensured that only papers directly relevant to the study objectives and providing meaningful insights were considered. In particular, **IC1** ensured that the papers were directly relevant to our study by explicitly addressing non-functional requirements within the specific context of ML-enabled systems. By requiring the use of specific non-functional requirements terminology, we filtered out papers that might discuss non-functional requirements in a more general or unrelated context, thus maintaining a focused and relevant selection. With **IC2**, we ensured that the selected papers provided a substantial and focused discussion on specific non-functional requirements. It prevented the inclusion of papers that only briefly mentioned non-functional requirements without delving into detailed analysis or discussion. This way, we ensured that the papers contributed meaningful insights and significant knowledge about specific non-functional requirements in ML-enabled systems. By including **IC3**, we ensured that the selected papers addressed the challenges associated with the management of non-functional requirements. This criterion allowed us to gather comprehensive insights into the difficulties and obstacles faced when dealing with non-functional requirements in ML-enabled systems, thus enriching our understanding of the practical implications and complexities involved. Finally, **IC4** ensured that the context in which the non-functional requirements were studied was clearly defined, allowing us to understand how non-functional requirements were addressed across different domains. Whether the paper addressed non-functional requirements in a general sense or within a specific domain, this information was crucial for understanding the applicability of non-functional requirements in various real-world scenarios.

The application of the exclusion and inclusion criteria ensured that only papers explicitly addressing NFRs within the context of ML-enabled systems were considered. In this regard, it is worth clarifying how the criteria concerning the evaluation of whether a paper revolved around NFRs (e.g., **EC10**) were practically applied. In the first place, we defined an NFR as a *"quality attribute or property of a system that specifies criteria to evaluate its operation, performance, or other overarching attributes, rather than its functional behaviors"* [12]. In the context of this study, NFRs were identified based on the following characteristics:

- **Explicit Terminology:** Papers that explicitly used terms such as "non-functional requirement" "quality attribute" or specific known NFRs like fairness, robustness, or security were included by our search.
- **Implicit Identification:** In cases where explicit terminology was absent, we relied on descriptions of system qualities or properties that aligned with widely recognized definitions of NFRs (e.g., qualities defined in ISO/IEC 25010 or by previous NFRs literature [23, 24, 27]). For instance, mentions of "bias reduction" or "system robustness" were considered relevant to fairness or robustness, respectively.
- **Distinction from Functional Requirements:** Attributes describing how a system operates or performs (e.g., low latency, energy efficiency) were classified as NFRs. In contrast, functional requirements, which define what the system does (e.g., an ML system's ability to classify images), were excluded.

Specifically, the characteristics described in the papers were evaluated based on their alignment with system qualities or attributes rather than solely focusing on the system's functional behaviors. It was insufficient for a quality to simply be described using general terms or adjectives like *"efficient"*, *"fast"*, or *"reliable"*. Instead, the quality had to represent a broader, overarching attribute that contributes to the system's operation, performance, or non-functional goals. For instance, terms such as *"efficient"* were considered relevant only when tied to system-level goals like *"low latency"* or *"scalability"*, namely qualities that fit within the framework of NFRs.

3.3. Research Method to Conduct the Seed Set Search

As further detailed in Section 3.7, the research method implemented through the guidelines by Kitchenham et al. [33] was required to be integrated with additional steps to ensure completeness. When executing the search string, we realized that a number of relevant primary studies did not refer to the general concept of non-functional requirements but rather preferred to point to the specific aspect pertinent to their particular area of study. For example, the article by Chen et al. [S1] proposed to address fairness and performance attributes of ML-enabled systems: in the content of the paper, the authors did not refer to the term *"non-functional requirements"*, even though they actually dealt with them. This example well explains the limitations observed: more particularly, it is not a limitation of the Kitchenham et al.'s guidelines per-se, but rather the challenge lies in the *inherent variability in terminology* used by different authors. Consequently, relying strictly on the term *"non-functional requirements"* in our search strategy would have led to the omission of significant studies that addressed non-functional aspects under different terminologies. This limitation highlights the need for a flexible and adaptive search strategy that can accommodate the diverse ways in which non-functional requirements are described in the literature - this is the reason why we opted for a hybrid search procedure that combines multiple search approaches.

Table 1: Conferences and journals considered in the scope of the seed set identification process.

Venue	Type	Name	Ranking
Journal	SE-related	*IEEE Transactions on Software Engineering* ***(TSE)***.	**Q1**
Journal	SE-related	*ACM Transactions on Software Engineering and Methodology* ***(TOSEM)***.	**Q1**
Journal	SE-related	*Empirical Software Engineering* ***(EMSE)***.	**Q1**
Journal	SE-related	*Journal of Systems and Software* ***(JSS)***.	**Q1**
Journal	SE-related	*Information and Software Technology* ***(IST)***.	**Q1**
Journal	AI-related	*Journal of Engineering Applications of Artificial Intelligence* ***(EAAI)***.	**Q1**
Journal	AI-related	*IEEE Transactions on Knowledge and Data Engineering* ***(TKDE)***.	**Q1**
Journal	AI-related	*IEEE Transactions on Pattern Analysis and Machine Intelligence* ***(TPAMI)***.	**Q1**
Journal	AI-related	*IEEE Transactions on Neural Networks and Learning Systems* ***(TNNLS)***.	**Q1**
Conference	SE-related	*IEEE/ACM International Conference on Software Engineering* ***(ICSE)***.	**A***
Conference	SE-related	*Joint European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering* ***(ESEC/FSE)***.	**A***
Conference	SE-related	*IEEE/ACM Automated Software Engineering Conference* ***(ASE)***.	**A***
Conference	SE-related	*IEEE International Conference on Software Maintenance and Evolution* ***(ICSME)***.	**A**
Conference	SE-related	*IEEE International Conference on Software Testing, Verification and Validation* ***(ICST)***.	**A**
Conference	SE-related	*ACM SIGSOFT International Symposium on Software Testing and Analysis* ***(ISSTA)***.	**A**
Conference	SE-related	*IEEE International Requirements Engineering Conference* ***(RE)***.	**A**
Conference	SE-related	*IEEE International Conference on Software Analysis, Evolution and Reengineering* ***(SANER)***.	**A**
Conference	SE-related	*IEEE International Conference on Program Comprehension* ***(ICPC)***.	**A**
Conference	SE-related	*IEEE International Working Conference on Mining Software Repositories* ***(MSR)***.	**A**
Conference	AI-related	*The Association for the Advancement of Artificial Intelligence* ***(AAAI)***.	**A***
Conference	AI-related	*International Joint Conference on Artificial Intelligence* ***(ICAI)***.	**A***
Conference	AI-related	*Empirical Methods in Natural Language Processing* ***(EMNLP)***.	**A***

To tackle this issue, we employed a method known as *seed set identification*, where we systematically examined research papers from top conferences and journals within software engineering and artificial intelligence. This approach, as highlighted by Wohlin et al. [74], complements systematic literature reviews when standard guidelines fall short. We focused on both software engineering and artificial intelligence venues, considering that primary studies might be found in either domain. For software engineering venues, we identified relevant conferences and journals using the *CORE* Ranking system for conferences (we only considered A* and A conferences) and the *Scimago* Journal Ranking for journals in the *'Software'* category (we only considered journals falling into the first quartile (Q1)). We finally selected the conferences and journals marked as *'SE-related'* presented in Table 1. Subsequently, we reviewed papers published in these venues from 2023 back to 2012. As a final step, we analyzed each conference website, and for journals, we searched for papers via the DBLP,[8] the most extensive computer science bibliography library. We applied the same exclusion and inclusion criteria defined for the search process conducted using the standard guidelines (see Section 3.2.3).

The decision to limit to A* and A conferences and Q1 journals stems from the consideration

[8]The *DBLP*: `https://dblp.org/`.

that these venues are widely recognized by the research community as highly relevant and high-quality. These venues are associated with rigorous review processes and high impact, ensuring that the primary studies selected meet a recognized standard of excellence and relevance. At the same time, we acknowledge that our approach may have excluded some relevant studies from other reputable conferences and journals, e.g., REFSQ.[9] This remains a limitation of our systematic literature review and is, in fact, a limitation shared by any systematic literature review. As for the focus of our work, there are three considerations to make. In the first place, our seed search included the major conference on requirements engineering (RE) in an effort to capture key research contributions from the area of requirements engineering. In the second place, many of the other conferences, e.g., ICSE, FSE, ASE, and journals considered, e.g., TSE, TOSEM, regularly publish research on requirements engineering, hence ensuring that our review encompassed a broad and representative sample of the most influential work on non-functional requirements of ML-enabled systems. Finally, research on non-functional requirements in ML-enabled systems often intersects with various domains within software engineering, such as machine learning, system architecture, and human-computer interaction. Consequently, such research may frequently be published in venues that cover these broader or intersecting areas, where there is a wider audience that might benefit from these findings. The breakdown of our primary studies (which is discussed as part of Section 4) seems to prove our considerations. Furthermore, vision papers and survey studies, such as the ones by Köhl et al. [34], and Habibullah et al. [24], highlighted the early stages of this research area and the ongoing efforts to build a comprehensive understanding of these requirements.

Expanding the search to include all relevant artificial intelligence venues would have been excessively labor-intensive. Applying the same criteria used for software engineering, we would have identified around 70 Q1 journals and over 40 A* and A conferences to review from 2023 to 2012. This would have resulted in an estimated 200,000 potential hits for analysis, deemed impractical for manual assessment. As such, we were required to identify solutions to balance the thoroughness of the search process with its practical feasibility. To ensure a comprehensive systematic literature process, we focused the seed search on artificial intelligence venues that are more likely to publish engineering or empirical approaches relevant to our research questions.

We first identified an online repository, named *'AI Venues'*,[10], which lists the whole set of artificial intelligence conferences and journals along with their ranking and H-index. We then associated to each journal the corresponding rank provided by *Scimago* and to each conference the rank provided by *CORE*. Besides discarding the venues that did not meet our selection criteria (rank=A* or A for conferences, rank=Q1 for journals), we filtered out the venues that revolved around too specific techniques or technologies, e.g., the *IEEE Transactions on Image Processing* journal, and favor instead the venues that encompassed a broader spectrum of engineering or empirical approaches applied for the development of ML-enabled systems, e.g., the *IEEE Transactions on Neural Networks and Learning Systems*. We selected four Q1 journals and three A* conferences whose themes were either related to improving AI approaches or using engineering or empirical approaches to AI. This process led to selecting the venues marked as *'AI-related'* reported in Table 1. We scanned all the papers published to these venues between 2012 and 2022, applying the same selection process described in Section 3.2.3.

[9]The REFSQ conference: `https://2024.refsq.org/series/refsq`.
[10]The *'AI Venues'* repository: `https://aivenues.github.io/`

3.4. Research Method to Conduct the Snowballing Process

The third step revolved around the forward and backward snowballing. This is the procedure through which a researcher systematically scans the ingoing and outgoing references of the primary studies with the aim of identifying new relevant resources to address the research questions [72]. By "ingoing references", we meant references from other papers that cite the primary study, i.e., other articles and papers that have cited the primary study in their own bibliographies. By "outgoing references", we meant references cited by a primary study, i.e., the list of articles and papers that the primary study has referenced in its bibliography.

In our case, the primary studies identified as a consequence of applying the exclusion and inclusion criteria on the studies retrieved using the standard guidelines and the seed search were scanned. In particular, we applied an iterative procedure in which all ingoing and outgoing references of the primary studies were first considered. Afterward, we reiterated the procedure for the newly acquired studies in an effort to identify additional resources. Overall, four rounds of backward and four rounds of forward were conducted: we stopped at four as we reached saturation, namely, we did not identify any additional papers to include. From a practical perspective, the snowballing steps were conducted through the use of *Google Scholar*,[11] an academic search engine which simplifies the analysis of ingoing and outgoing references of research papers. This was the only step in verifying whether the papers cited or citing the primary resources were published. Whenever needed, i.e., whenever we identified a new paper that was not previously identified through the initial or the seed search, we searched the title of the paper on *ACM Digital Library*, *Scopus*, and *IEEEXplore* to verify its publishing status. If the paper was published, it was accepted for the subsequent steps of our research method.

3.5. Research Method to Conduct the Quality Assessment

Once we had completed the application of the three complementary search processes described in the previous sections, we conducted a quality assessment of the resources that successfully passed the inclusion criteria. The implementation of the quality assessment process started with the definition of qualitative questions aiming at operationalizing the main pieces of information that a primary study should have had to be useful to address our research questions. These were defined according to the research objectives initially defined:

- **Q1:** *Does the paper provide a detailed and clear definition or discussion of the non-functional requirements?* This should include an in-depth explanation, context, and significance of the non-functional requirements within ML-enabled systems.
- **Q2:** *Does the paper offer specific examples, case studies, or empirical data related to the non-functional requirements and their challenges?* The paper should provide concrete instances, applications, or data supporting the discussion of non-functional requirements and their associated challenges.
- **Q3:** *Does the paper discuss how the non-functional requirements apply within general-purpose or specific domains?* This may include the relevance of the domain to the non-functional requirements, specific challenges within the domain, or any domain-specific considerations or solutions.

[11] Link to *Google Scholar*: `https://scholar.google.com/`

The quality assessment criteria are designed to ensure that the selected papers provide a thorough, evidence-based, and context-aware discussion of non-functional requirements in ML-enabled systems. **Q1** ensures clarity and depth in defining and discussing the non-functional requirements, establishing a solid foundation for the study. **Q2** emphasizes the importance of practical examples and empirical data, which bolster the paper's arguments and demonstrate real-world relevance. **Q3** focuses on the applicability of non-functional requirements across different domains, ensuring that the discussion is comprehensive and considers domain-specific challenges and solutions. These criteria collectively ensure that the included papers contribute significantly to understanding and addressing non-functional requirements in ML-enabled systems. In particular, **Q1** and **Q3** were used to address $\mathbf{RQ}_1$, while **Q2** and **Q3** to address $\mathbf{RQ}_2$.

When assessing the primary studies against each of the qualitative questions, previous systematic literature reviews (e.g., [2, 6, 16]) assigned a boolean value to indicate whether a study had or not the quality required with respect to a property considered. However, assessing the primary studies through boolean values might be challenging, other than possibly threatening the validity of the assessment. For instance, there might be cases where the challenges associated with non-functional requirements may be logically elicited from the text, even though not explicitly stated. To deal with this process, we, therefore, opted for the application of a *fuzzy linguistic approach* [3], which consists of rating each primary study through a continuous variable ranging between 0 and 1. In particular, the scores were assigned as follows:

- 0 ⇒ **No**
- 0.1-0.3 ⇒ **Rarely**
- 0.4-0.6 ⇒ **Partly**
- 0.7-0.9 ⇒ **Mostly**
- 1 ⇒ **Yes**

In other terms, for each qualitative question, the primary studies were assigned a value reporting how explicit and clear the content was in that respect. At the end of the evaluation conducted for each question, the total merit of a primary study was computed as follows:

- 0 ⇒ **No**
- 0.1-1.1 ⇒ **Rarely**
- 1.2-2 ⇒ **Partly**
- 2.1-2.7 ⇒ **Mostly**
- 2.8+ ⇒ **Yes**

To be finally accepted as part of our systematic analysis, the primary study should have obtained a final score of more than 1.6, i.e., it should have partly specified the required pieces of information to address the research questions of the study.

3.6. Design of the Data Extraction Form

As a final step of the research method applied to address the goals of our study, we designed the data extraction form, namely the specification of the pieces of information to collect when addressing our research questions. Table 2 summarizes the data collected, reporting (i) the dimension the attribute group referred to, (ii) the scope where the data has been used, (iii) the description of the dimension considered, and (iv) the specific attributes considered. As shown in the table, we collected fuor main categories of information. At first, we identified pieces of information that might help statistically describe our sample in terms of bibliometrics, e.g., publication year and venues: we used the knowledge acquired to describe the trends in terms of publication, the most relevant venues accepting research papers on non-functional requirements of ML-enabled systems. Besides these meta-data, we then collected and stored information that may be directly connected to the specific research questions posed in our study and that, therefore, was used in the context of the data analysis and reporting process.

As for $\mathbf{RQ}_1$, we collected multiple pieces of information about the non-functional requirements treated in the primary studies. In particular, we collected the specific 'name' of a requirement, its 'description', and the 'domain' where it has been analyzed. As for the 'domain' information, whenever an explicit reference to a certain domain was available, we classified the domain accordingly. In the cases where the domain was not explicitly mentioned, we categorized the non-functional requirements under *"General Environment"*. This classification was used when the primary study discussed the non-functional requirement in a non-specific context. Therefore, *"General Environment"* is not a specific environment per se, but a category we used to denote non-functional requirements that were addressed in a general, non-domain-specific manner but that can be considered broadly applicable across any kind of domain. As for $\mathbf{RQ}_2$, we collected information about the challenges described in the literature, considering their 'nature', 'causes', and 'impact'. The attributes extracted in this stage were used as part of the data analysis processes employed to address each research question.

Table 2: Data extraction Form.

Dimension	Scope	Description	Attribute Collected
Paper Information	**Bibliometrics**	This component includes general information and criteria used to assess the quality of the selected studies.	Seed set or Systematic Literature Review
			Year of publication
			Accepted Score
			Journal or a conference.
			Venue
Terms concerning non-functional requirements	$\mathbf{RQ}_1$	This component encompasses all the keywords that make it possible to describe non-functional requirements, both those that already exist in the literature and new non-functional requirements that emerged from this study.	Non-Functional Requirements
Domains where non-functional requirements were studied	$\mathbf{RQ}_1$ and $\mathbf{RQ}_2$	This component includes the domains in which non-functional requirements have been studied and found to be problematic.	ML Domains
Challenges of SE approaches for ML-enabled systems	$\mathbf{RQ}_2$	This component contains a list of challenges explicitly stated in primary studies and any future challenges and problems emerging from these studies. This information will help us identify areas where further research is needed to improve SE approaches for ML-enabled systems and improve the existing ones.	Challenges
			New Challenges Emerged

3.7. Execution of the Research Methods

Figure 1 presents the outcomes of our research methods. The systematic literature review commenced on June 26, 2024, yielding a total of 3,054 hits, predominantly from *Scopus* (2,334), with fewer from *ACM Digital Library* and *IEEEXplore* (662 and 58, respectively).

Upon execution of the search string and the seed search, we imported the resulting articles in a shared Excel sheet. At first, the duplicated articles (coming as a result of the search string execution against different databases) were removed, with only one instance retained. Through the meta-data provided by the databases, we could automatically filter out the papers not written in English, short papers, papers not published/peer-reviewed, papers that did not fall into themes of computer science and computer engineering, and papers published before 2012: this process reduced the number of papers to 1,916. Next, each remaining paper was manually scanned to apply the additional exclusion criteria. We verified that each paper was actually accessible, meaning it was available for full-text review. If a paper was not accessible, it was excluded.

For the remaining papers, we analyzed (1) the venue where the paper was published, (2) the article type, and (3) the topic addressed. Regarding the venue, we excluded workshop papers. For the article type, we excluded vision papers, systematic literature reviews, and literature surveys by examining the title, keywords, and, when necessary, the abstract. To verify the scope of

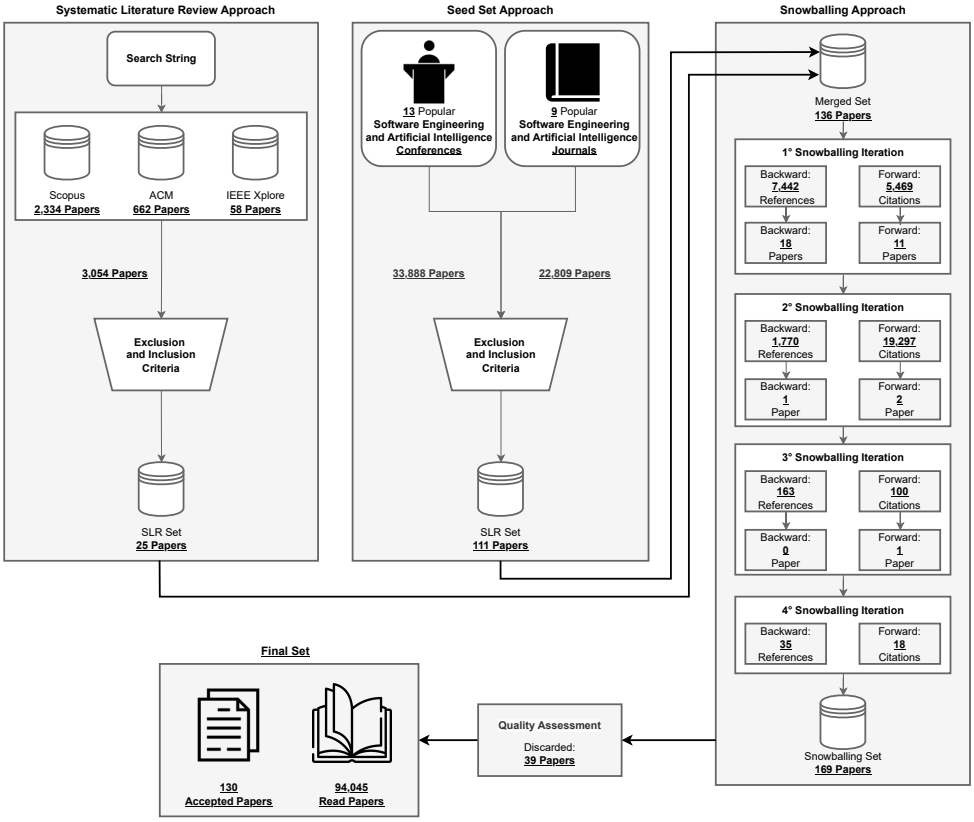

Figure 1: Overview of the process for selecting papers.

the article, we relied on the title and keywords, and read the abstract, introduction, and conclusions when additional clarification was needed. This scanning also allowed us to identify journal articles that extended conference papers, leading to the exclusion of the conference proceedings. In addition, we excluded papers based on whether they addressed any non-functional requirement (**EC10**). As explained in Section 3.2.3, papers were analyzed with respect to the extent to which they discussed quality attributes in relation to the system's operation, performance, or non-functional goals. As the reader may expect, attributes such as classification accuracy and performance were widely discussed in the considered literature; however, not all papers explicitly or implicitly contextualized these attributes as non-functional requirements. According to our analysis, many papers address these attributes solely in the context of functional goals, such as improving prediction accuracy or optimizing model performance for a specific task, without considering their role as overarching system-level qualities or linking them to the broader concept of NFRs. An example is the paper by Suwonchoochit and Senivongse [65]. The primary focus of the paper is on improving prediction accuracy through algorithmic optimization without addressing accuracy as a broader system-level quality or linking it to non-functional requirements. Specifically, the paper employed various ML algorithms, with Extra Trees and TF-IDF emerging

as top-performing models to classify user-reported database-related issues on Stack Overflow, but it does not consider accuracy in terms of trade-offs with other system qualities (e.g., fairness or robustness) or its implications for system deployment and operation. Consequently, while accuracy was a central theme, it was not treated as a system-level non-functional requirement. This lack of contextualization within the broader framework of non-functional requirements led to its exclusion based on **EC10**. On the contrary, let us consider the paper by Biswas and Rajan [S2]. This paper was not excluded in this phase - it is actually one of those finally included in our work. The paper explicitly framed accuracy as a system-level quality and linked it to broader non-functional requirements. Indeed, the paper explores fairness in ML models sourced from Kaggle. It aims to evaluate bias in these models, investigate its causes, and analyze the effectiveness and trade-offs of various bias mitigation techniques. By presenting accuracy as a non-functional requirement with system-level implications, the paper was not excluded and led to the following phase, which was concerned with the application of the inclusion criteria.

After completing the exclusion criteria, we proceeded with the inclusion criteria. While the application of these two sets of criteria may be performed simultaneously, we chose to separate them to ensure a thorough and systematic review process. This approach allowed us to focus first on eliminating papers that clearly did not meet our basic requirements before concentrating on those that potentially fit our study's scope. The inclusion criteria were first assessed by scanning the title, keywords, abstract, introduction, and conclusions of the papers. In the case these pieces of information were not sufficient to establish the suitability of a paper, a full read-through of the entire paper was conducted. The vast majority of the papers analyzed in this stage were removed as a consequence of **IC1** and **IC2**, as they lacked focus on ML-enabled systems or they tangentially mentioned non-functional requirements. According to our findings, most of the papers mentioning terms such as *"Non-Functional Requirement"*, *"Non Functional Requirement"*, and *"NFR"* focused on non-functional requirements of classical software systems, i.e., systems that do not present any machine learning component. Furthermore, most papers addressing specific aspects that could, in principle, be connected to ML-enabled systems, e.g., adversarial sampling, did not explicitly link these aspects to specific non-functional requirements. These papers often lacked the detailed terminology or focused analysis necessary to contextualize their findings within the framework of non-functional requirements in ML-enabled systems. For instance, the paper by Nguyen et al. [52] represents a valid example that explains the impact of **IC1** and **IC2** to assess the relevance of the primary studies considered in our work. The article proposed an adversarial sampling technique to generate training samples for API recommendation systems. However, the paper did not explicitly refer to any specific non-functional requirements in their reporting. Instead, they focused primarily on the technical aspects of the proposed technique without addressing the broader non-functional requirements, such as reliability or security, that this technique could impact. This omission limits the applicability of their findings in the scope of our work, as it prevents a clear understanding of how the proposed technique addresses or impacts specific non-functional requirements critical to ML-enabled systems. As a consequence, it could not be included in our review.

The remaining papers satisfied **IC3**. As for **IC4**, papers that satisfied this criterion either defined a general framework for addressing NFRs across ML-enabled systems or provided a detailed analysis of NFRs within a specific domain. These papers offered insights into how NFRs were handled, making them relevant to the study. As an example, let's consider the paper by Aminifar et al. [S3]. This article provided a general-purpose framework to optimize accuracy and privacy, which would be applicable to any ML-enabled systems, independently from the specific application domain. Similarly, the paper by Hutiri et al. [S4] explores the emergence

and propagation of biases in on-device ML workflows, emphasizing the impact of design choices made during development. On-device ML, commonly deployed on resource-constrained IoT devices, may impact privacy, energy Consumption, and fairness. Both these papers were included in the study, as they properly reflected the intent of **IC4**. On the contrary, papers that did not satisfy **IC4** typically mentioned NFRs superficially without grounding them in any meaningful context. For instance, a paper that briefly referred to "security" without explaining how it applies to a general or specific domain was excluded.

Overall, the application of the exclusion and inclusion criteria led to the exclusion of 3,054 papers, hence leading to 25 primary studies that were employed in the context of the snowballing process. This latter step was conducted by scanning the titles of the papers with ingoing or outgoing references to the primary studies; the papers that were deemed potentially relevant were then subject to the inclusion criteria to establish their suitability for the study.

The 25 papers passing the inclusion criteria were then subjected to quality assessment, following the procedure described in Section 3.5. As an outcome, 11 papers were finally considered for inclusion in the study: these papers provided detailed discussions, comprehensive analyses, and significant insights into non-functional requirements in ML-enabled systems, thereby contributing meaningfully to our understanding of the topic.

At the same time, the limited amount of relevant papers, due to the *inherent variability in terminology* used in relevant studies, convinced us to augment the search process with additional steps. We, therefore, conducted the (1) seed set search described in Section 3.3, which led to the identification of 111 additional primary studies; and (2) the snowballing search described in Section 3.4, which contributed with 136 primary studies more. To be included, the papers coming from these two additional search steps had to satisfy both the exclusion and inclusion criteria, with key discriminators being **IC1** and **IC2**. Specifically, 99.8% of the papers were excluded because they did not explicitly link to non-functional requirements within the title, keywords, abstract, introduction, and conclusions. This stringent application of the criteria was required to consider a manageable amount of resources and ensured that only papers with a clear focus on non-functional requirements in ML-enabled systems were considered, thereby maintaining the relevance and quality of the primary studies included in our review.

The set of 169 papers was then analyzed with a quality assessment process on the full text. The majority of these papers were excluded due to not meeting the thresholds for **Q1** and **Q2**. While many papers scored well on **Q3**, discussing the domain-specific or general-purpose applicability of non-functional requirements, they lacked either a clear and detailed definition of NFRs or robust examples and empirical data to substantiate their claims. The application of these criteria led to the exclusion of 39 articles from the initial set. Summing up the primary studies identified by using the three different methods, our hybrid systematic literature review considered a total of 130 papers, which were finally used to address our research questions.

From an operational perspective, the procedures described above were primarily conducted by the first author of the paper. However, to ensure robustness and soundness, we implemented a validation phase in which we verified that the papers scanned throughout the process were accurately assessed for their suitability for the study. More particularly, from the initial set of 3,054 papers extracted through the original search string, i.e., the set of papers coming from the methodological steps presented in Section 3.2, we retained a sample of 100 papers to be used as a validation set. Both authors independently reviewed the papers in the validation set, applying the exclusion/inclusion criteria and quality assessment. They then compared their evaluations on the individual criteria, finding agreement in 87% of the cases. For the remaining discrepancies, they engaged in discussions to clarify the rationale behind their differing opinions and reach a

consensus. Based on the outcome of the discussion, they decided to retain an additional set of 50 papers for a second round of comparison. Although the initial level of agreement was considered sufficiently high, this second round was conducted to ensure that the previous discussions led to common evaluation criteria. This step was crucial to verify that the divergences observed in the first round had been effectively resolved and that both authors were consistently applying the exclusion/inclusion criteria and quality assessment. This additional round resulted in a full agreement between the authors, thereby establishing the foundation for the first author to proceed with the assessment of the entire set of papers. To make the reader aware of the effort required to conduct the search/analysis process and contribute to the transparency/replicability of our work, we also estimated the number of person-hours invested in the work. As for the first author, the estimation is about 850 person-hours; as for the second, it is about 400 person-hours.

4. Analysis of the Results

In this section, we report quantitative and qualitative insights coming from the data extraction and analysis phase. We first analyze bibliometric data to describe the sample considered in our research. Afterward, we address the specific research questions of the study.

4.1. Bibliometrics

From the total of 130 primary studies included, most of them (63, 48%) received an overall quality score of *"Mostly"*, i.e., they specified the pieces of information required to address our research questions mostly explicitly. The other 67 primary studies were more implicit and, indeed, reached an overall quality score of *"Yes"* or *"Partly"*. The primary studies were mostly retrieved through the seed set search (91, 70%), while the other 27 resources (21%) were identified by snowballing the seed primary studies identified. Only 11 papers were retrieved through the systematic literature search and only one additional resource could be identified through the snowballing process conducted on the set of papers identified with the systematic search. These considerations further justified our choice of complementing the traditional systematic literature search with additional instruments.

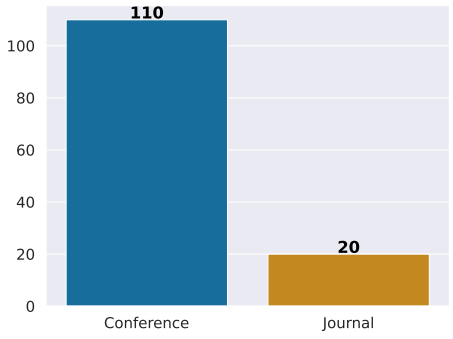

Figure 2: Papers published per publication venue.

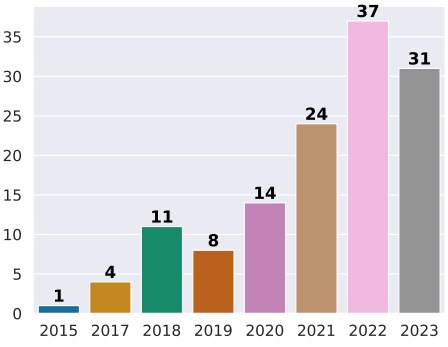

Figure 3: Papers published on the matter over time.

More interesting were the insights coming from the publication venues, which are shown in Figure 2. We observed that the vast majority of the primary studies (110, 85%) appeared in conference proceedings, with a limited amount of resources published in journals (20, 15%).

Figures 4 and 5 report the breakdown of the publications by venue. From these figures, we may provide additional observations on the state of the research in non-functional requirements of ML-enabled systems. Surprisingly, only a limited amount of primary studies (2) come from the major requirements engineering conference (RE). Some considerations may explain this result. First, the emerging nature of the topic implies that much of the foundational and exploratory work is still being conducted, often appearing first in venues focused on broader software engineering conferences. In addition, research on non-functional requirements in ML-enabled systems often intersects with various domains within software engineering, such as machine learning, system architecture, and human-computer interaction. Consequently, such research may frequently be published in venues that cover these broader or intersecting areas, where there is a wider audience that might benefit from these findings. The breakdown of our primary studies seems to prove our considerations - the most popular venues are ICSE and ESEC/FSE for conferences and TOSEM for journals. Furthermore, vision papers and survey studies, such as the ones by Köhl et al. [34], and Habibullah et al. [23], highlight the early stages of this research area and the ongoing efforts to build a comprehensive understanding of these requirements.

The distribution of primary studies across conferences and journals also suggest that the research is still at its early stage, with researchers interested in discussing recent advances in venues that allow discussion and interaction with the research community, i.e., conferences. The analysis of the amount of papers published on the matter per year confirmed the early nature of the research area (Figure 3). According to our findings, until 2015, no articles were published, and from 2015 to 2018, only a few papers were published, while a notably increasing trend could be found in the last three years. Performing a systematic synthesis of the knowledge collected so far could provide a relevant boost to the research activities that will be performed in the next years: our work indeed identifies multiple implications and challenges that the software engineering research community will be called to address.

4.2. $\boldsymbol{RQ}_1$ - *What are the non-functional requirements of ML-enabled software considered by researchers?*

The first goal of our work was to classify the non-functional requirements of machine learning-enabled systems. In the following section, we report the data synthesis process and the results achieved.

Data Analysis and Synthesis. Based on the data collected through the data extraction form, we addressed $\mathbf{RQ}_1$ by applying a three-step systematic classification exercise through which we (i) elicited the non-functional attributes that they aimed to address, (ii) characterized the non-functional requirements, and (iii) grouped them according to two main classification criteria. In particular, the three steps were conducted as follows:

- *Extraction of Non-Functional Requirements.* We analyzed the primary studies to identify and extract non-functional requirements explicitly mentioned in the papers. This step involved thorough reading and annotation of the relevant sections discussing the non-functional requirements. Whenever possible, i.e., when considering the papers that received a quality score of "Yes", the extraction was relatively straightforward as these resources explicitly referred to the non-functional requirements considered. In the other cases, relevant terms connected to non-functional aspects were extracted and elaborated for consistency. For example, the primary study [S5] reported terms like *"memory problems"* and *"battery drain"*, which we used to interpret the context of the study and mapped them to the non-functional requirement named 'energy consumption'.

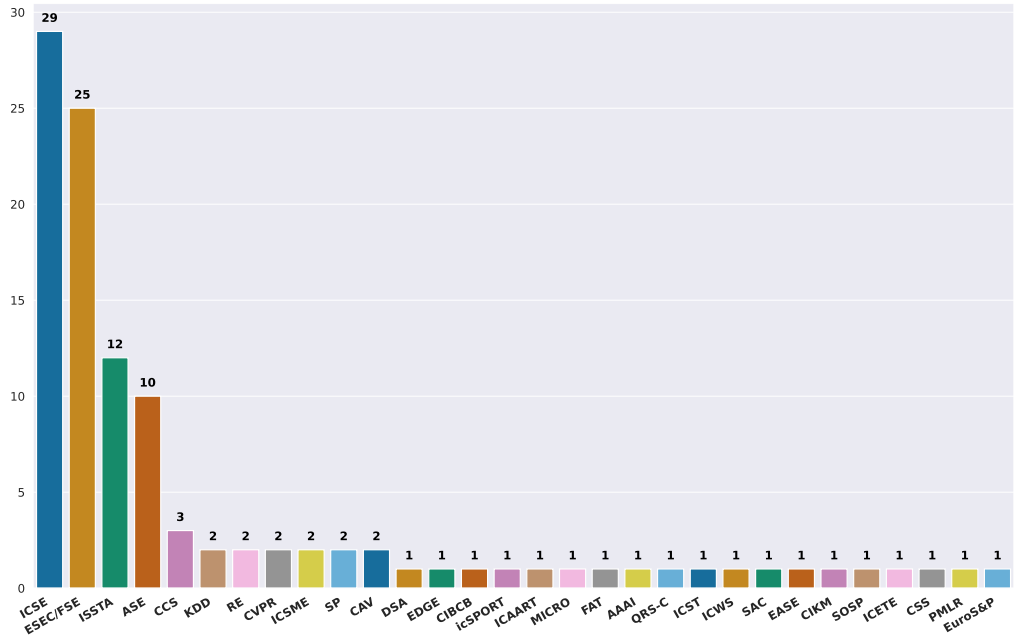

Figure 4: Conference Venues by the number of papers

- *Naming and Describing Non-Functional Requirements.* Each extracted non-functional requirement was assigned a unique name and a detailed description. In this stage, possible naming inconsistencies that emerged in the previous step were solved. The description was intended to capture the essence and scope of the requirement as discussed in the primary studies.

- *Mapping to Reference Class.* The non-functional requirements were then mapped to high-level reference classes based on two classification criteria: (1) the non-functional requirements must have a significant impact on the high-level reference class; (2) non-functional requirements sharing similar purposes or characteristics must be grouped together under a common reference class. For example, non-functional requirements like privacy and security were grouped under the 'Resiliency' class because they both contribute to the system's ability to handle unexpected events and threats.

The names assigned to the classes, the description of each non-functional requirement, and the classification exercise as a whole, were informed by different sources of the existing body of knowledge, i.e., by the primary studies collected in this paper, other resources of the state of the art not directly related to our research goals, and the *Systems & Software Quality Requirements & Evaluation* (ISO/EIC 25010).

The process was initially conducted by the first author of the paper, who (1) analyzed the primary studies to elicit the non-functional attributes considered and (2) provided a preliminary version of the taxonomy. This preliminary taxonomy aimed to homogenize definitions and classifications by leveraging existing knowledge and favoring well-established class names whenever possible, thus providing a more comprehensive and usable taxonomy. The preliminary taxonomy

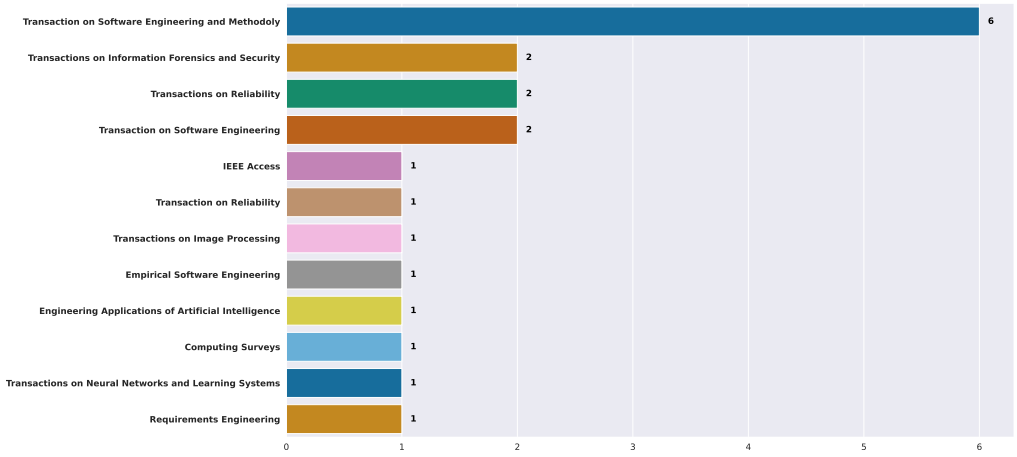

Figure 5: Journal Venues by the number of papers

was later subject to evaluation. The second author joined the process at this stage, and a discussion was opened on the produced taxonomy. The two authors discussed the consistency of the classification, as well as the names and descriptions assigned. Modifications were made where necessary—this occurred in four cases, where some non-functional requirements were grouped differently, and the names assigned to the classes were adjusted.

Addressing RQ$_1$. In the first place, as a result of this procedure we could identify a total amount of 31 non-functional requirements. These are shown in Table 3 along with the reference to the primary studies identifying or defining them. Interestingly, our systematic exercise could identify not only non-functional requirements that emerged already in previous studies [23, 24, 27], but also additional categories that were not previously pointed out, i.e., *transferability*, *cost*, *accountability*, *energy consumption*, *sustainability*, *capacity*, *stability*, *behavioral*, *imperceptibility*, *efficiency*, *availability* and *adaptability*.

The classification exercise led to the identification of six main classes of non-functional requirements: Table 4 reports, for each class, the set of non-functional requirements belonging to the class, along with their definition and meaning. The column 'Meaning' reports three symbols explaining whether each non-functional requirement can be considered (1) new with respect to traditional non-functional requirements previously investigated ('New'), (2) not new but with altered meaning ('Altered'), or (3) not new but with the same meaning ('Same'). More specifically, we identified the following classes:

Accuracy. The first class contained only the accuracy requirement, namely the degree to which a model's predictions match the actual values. This represents a new category of non-functional requirement that is specific to ML-enabled systems. We considered accuracy as a non-functional property for two main reasons. In the first place, multiple papers included in our systematic literature review, e.g., [23, 24, 27], described accuracy as a non-functional requirement, defining it as the number of correctly predicted data points out of all the data points. As our work aims at synthesizing the current knowledge available on the matter, we preferred to be conservative and considered accuracy as reported in the state of the art. In the second place, the definition is in line with the concept of non-functional requirement, i.e., *"a condition that specifies a crite-*

Table 3: Full list of the classified non-functional requirements in ML-enabled systems, along with the reference to the corresponding primary studies.

NFRs	Resources	References
Accuracy	104	[S1–S4, S6–S105]
Robustness	65	[S7, S10, S14, S17, S18, S21, S25–S29, S33, S35, S39–S41, S92, S94, S106–S115] [S1, S42–S47, S52, S55–S57, S59, S62, S75, S77, S80, S84, S85, S88, S89, S91, S116] [S63–S65, S68, S71, S73, S75, S77, S93, S95, S98, S99, S101, S103, S104, S117, S118]
Fairness	34	[S13, S19, S20, S22, S30, S36–S38, S48, S50, S107, S109, S114, S119–S123] [S1, S2, S4, S53, S64, S69, S76, S78, S79, S83, S86, S96, S102–S105]
Security	33	[S3, S14, S17, S23, S25, S27, S30–S33, S35, S39, S106, S110–S113, S115] [S40, S44–S47, S55, S58, S71, S73, S77, S98, S105, S116, S117, S124]
Performance	30	[S8, S11, S12, S14, S18, S21, S23, S28, S31, S37, S38, S106, S112, S113, S123] [S4, S5, S41–S43, S46, S49–S51, S57, S70, S81, S101, S117, S125]
Behavioral	21	[S10, S15, S24, S25, S32, S35, S41, S52, S107, S113, S114] [S61–S64, S70, S71, S84, S96, S103, S126]
Interpretability	20	[S9, S17, S30–S32, S40, S50, S60, S70, S72, S76, S78, S79, S113] [S82, S87, S105, S124, S127, S128]
Reliability	15	[S10, S30, S40, S41, S52, S93, S95, S97, S98, S100, S105, S113, S115, S118, S126]
Explainability	15	[S9, S31, S33, S53, S60, S70, S72, S87, S105, S114, S120, S124, S127–S129]
Safety	14	[S11, S24, S32, S35, S55, S56, S64, S82, S84, S98, S100, S105, S108, S118]
Cost	12	[S8, S18, S30, S45, S51, S52, S62, S67, S68, S82, S88, S130]
Retrainability	12	[S7, S18, S27, S47, S68, S76, S82, S91, S94, S105, S109, S115]
Energy Consumption	10	[S4, S5, S8, S12, S21, S26, S51, S100, S125, S130]
Capacity	10	[S4, S5, S8, S11, S81, S91, S94, S99, S101, S125]
Transferability	10	[S25, S26, S39, S55, S76, S89, S106, S108, S110, S112]
Privacy	9	[S1, S3, S23, S30, S34, S37, S86, S105, S115]
Transparency	8	[S31, S36, S105, S114, S122, S127–S129]
Reusability	6	[S16, S54, S90, S91, S94, S105]
Accountability	5	[S36, S119, S122, S127, S128]
Replaceability	5	[S16, S54, S90, S91, S94]
Scalability	5	[S40, S54, S84, S108, S111]
Sustainability	3	[S51, S99, S100]
Efficiency	3	[S21, S57, S105]
Stability	3	[S26, S40, S105]
Reproducibility	2	[S49, S105]
Maintainability	2	[S61, S105]
Usability	2	[S105, S129]
Imperceptibility	1	[S56]
Ethics	1	[S36]
Flexibility	1	[S115]
Availability	1	[S21]
Adaptability	1	[S26]
Traceability	1	[S105]

Table 4: Non-Functional Requirements in ML-enabled systems. The *'Meaning'* column reports whether the non-functional requirements are *'new'* (N), i.e., they are specific to ML-enabled systems, are *'altered'* (A), i.e., their meaning is different with respect to the definition of the non-functional requirements in non-ML-enabled systems, or are the *'same'* (S), i.e., their meaning is the same as the non-functional requirements in non-ML-enabled systems.

Cluster	NFRs	Meaning	Definition
Accuracy	Accuracy	N	The degree to which a model's predictions match the actual values.
Efficiency	Performance	S	The ability of a system to perform actions within defined time or throughput bounds.
	Capacity	A	The amount of space required to store the model and any associated data.
	Stability	N	Degree to which the output of a model varies as a consequence of perturbations to its input.
	Scalability	A	The capability to handle increased workloads by adding resources while maintaining or enhancing model performance.
Maintainability	Replaceability	N	The degree to which a model can be replaced or substituted with another model without significant changes to the system.
	Retrainability	N	The degree to which a model can be retrained on new data without significant performance loss.
	Reproducibility	N	The degree to which a model's results can be reproduced by others using different software or hardware.
	Transferability	N	The degree to which a model trained on one data set can be applied to another with similar characteristics.
	Reusability	A	The degree to which a model can be reused in different applications or contexts.
	Adaptability	A	The ability of the model to adapt to changing requirements or environments.
	Traceability	S	The ability to trace work items across the development lifecycle.
Resiliency	Security	A	The degree to which a model and its associated data are protected against unauthorized access, modification, or theft.
	Safety	A	The degree to which a model and its outcomes are safe for humans and the environment.
	Privacy	A	The degree to which a model and its associated data protect individuals' privacy rights and comply with data protection regulations.
	Robustness	A	The ability of a model to maintain its performance when faced with uncertainties or adversarial conditions.
	Reliability	A	Degree to which a model is resilient to errors and to variations of the surrounding environment.
	Behavioral	A	The degree to which a model's outcomes align with requirements and expectations.
	Flexibility	A	The degree to which a model can adapt to input data or environment changes without significant performance degradation.
	Availability	A	The degree to which a model is operational and accessible when needed, without significant downtime or interruption.
Sustainability	Fairness	N	The degree to which a model produces unbiased predictions and decision-making outcomes across different groups of individuals.
	Ethics	N	The degree to which a model mitigates potential societal risk.
	Accountability	N	The degree to which individuals or organizations are held responsible for the actions of the model and its outcomes.
	Cost	A	The overall economic means required to develop and maintain an ML-enabled system.
	Energy Consumption	A	The amount of energy required for training and inference of the model and its impact on the systems.
Usability	Interpretability	N	The degree to which a model's predictions and decision-making process can be explained in terms of causality or human-understandable concepts.
	Imperceptibility	S	The system's ability to produce outputs that are indistinguishable from what a human would produce in the same scenario.
	Explainability	A	The degree to which a model's predictions can be explained and understood by humans.
	Transparency	A	The degree to which a model's inner workings and decision-making process can be understood and evaluated by humans.

rion that may be used to judge the operation of a system rather than specific behaviors" [12]: as a matter of fact, accuracy does not define constraints on the specific functionality that a system should enable, but rather question how the system performs in terms of correctness of the predictions made. In other terms, accuracy represents an attribute that can be used to judge the operation of a system rather than its specific behavior, i.e., it is by definition a non-functional requirement. The class was made isolated because of two main reasons: (1) accuracy represents the key feature to optimize by ML-enabled systems; (2) the non-functional requirement cannot be conceptually compared to any other, being "unique".

Efficiency. This was concerned with the overall performance and effectiveness of ML-enabled systems. The ISO/IEC 25010 standard includes performance efficiency as one of its main characteristics, encompassing attributes such as time behavior, resource utilization, and capacity. As such, we exploited the same reasoning to cluster non-functional requirements such

as *performance*, *capacity*, *stability*, and *scalability*, under the *"Efficiency"* class. While the definition of *performance* is the same as the one available for non-ML-enabled systems, we classified a novel requirement, *stability*, and altered the meaning of *capacity* and *scalability*. More specifically, *stability* is a new non-functional requirement for ML-enabled systems due to the dynamic nature of machine learning models, which can undergo continuous updates and changes in response to new data. This dynamic aspect requires systems to maintain consistent performance and reliability over time, even as models evolve. The meanings of *capacity* and *scalability* are different for ML-enabled systems compared to traditional systems. In the context of ML-enabled systems, *capacity* not only refers to the system's ability to handle large volumes of data and numerous concurrent users but also encompasses the system's capability to manage the computational demands of inference tasks in terms of storage and deployment. This includes the ability to process complex models efficiently and to scale resources dynamically as computational requirements fluctuate. Similarly, *scalability* for ML-enabled systems involves more than just the traditional scaling of resources to accommodate increased load. It also includes the ability to scale machine learning models themselves, such as adjusting model complexity or deploying multiple models to handle different tasks. Scalability in ML-enabled systems must account for the need to distribute models across multiple nodes and ensure that inference can be performed efficiently across distributed environments. These additional layers of complexity necessitate a redefined understanding of capacity and scalability within the realm of ML-enabled systems.

Maintainability. Maintainability is chosen as a cluster to represent *replaceability*, *retrainability*, *reproducibility*, *transferability*, *reusability*, *traceability* and *adaptability*: it emphasizes the ability of a system to adapt and evolve over time. All these non-functional requirements are critical to ensure that the system can be maintained and updated as needed and that it can be easily adapted to new use cases or environments. All the non-functional requirements in this cluster, with the only exception of *traceability*, assume a different meaning with respect to non-ML-enabled systems. *Reusability* in ML-enabled systems involves the reuse of machine learning models and components in different systems. This includes the ability to transfer knowledge from one model to another, leveraging pre-trained models to reduce development time and improve performance in new applications. *Adaptability* refers to the system's ability to dynamically adjust to new data and changing conditions. In ML-enabled systems, this often means the ability to update models with new data to improve accuracy and relevance over time, as well as the capacity to integrate with new technologies and frameworks as they emerge. *Replaceability* is a new requirement that addresses the need to seamlessly replace or upgrade machine learning models and components without disrupting the overall system functionality. This is crucial in ML-enabled systems where models may need frequent updates or replacements to stay effective. *Retrainability* is another new requirement, emphasizing the system's capability to retrain models efficiently as new data becomes available. This ensures that the system remains accurate and up-to-date, reflecting the latest information and trends. *Reproducibility* is critical in ML-enabled systems to ensure that experiments and model training processes can be reliably repeated with the same results. This is essential for validating model performance and for iterative development. Finally, *transferability* refers to the system's ability to apply models and knowledge from one domain or task to another. This involves leveraging transfer learning techniques to adapt pre-trained models for new tasks, improving development efficiency and performance. By prioritizing maintainability, designers and developers can create reliable, efficient, adaptable, flexible, and sustainable systems over the long term.

Resiliency. Resilience represents *robustness*, *reliability*, *behavioral*, *flexibility*, *security*, *safety*, *privacy*, and *availability*, as it emphasizes the ability of a system to adapt and recover from adverse events while maintaining its functionality and performance. Robustness, reliability, and behavioral flexibility are essential to ensure the system can operate in various conditions and remain responsive to changing circumstances. Security, reliability, and confidentiality are also critical to ensure the system protects users and stakeholders from harm, including threats to their physical safety, personal information, and data confidentiality. In this cluster, the definition of all the non-functional requirements required to be adapted for ML-enabled systems. Specifically, *robustness* in ML-enabled systems refers to the system's ability to handle diverse and potentially unforeseen data inputs and scenarios without failing. This involves ensuring that models can generalize well to new, unseen data and are resistant to noise and adversarial inputs. *Reliability* is crucial for ML-enabled systems to maintain consistent performance over time. This includes the dependability of the model predictions and the ability to maintain performance despite variations in data quality or environmental conditions. *Behavioral* refers to the degree to which a model's outcomes align with the predefined requirements and expectations. This involves rigorous validation and testing to confirm that the model works as intended by the initial requirements. *Flexibility* refers to the system's capacity to adapt its behavior dynamically in response to changing environments and data patterns. For ML-enabled systems, this means models must be capable of adjusting their outputs on new inputs to remain effective. *Security* in the context of ML-enabled systems involves protecting the integrity of the model and data from malicious attacks. This includes safeguarding against attacks designed to manipulate model results and ensuring that the model or associated data is not stolen by attackers. *Safety* is a non-functional requirement that ensures the system does not cause harm to users or the environment. For ML-enabled systems, this involves ensuring that model decisions do not result in unsafe conditions, particularly in critical applications such as autonomous driving or healthcare. *Privacy* is critical in ML-enabled systems to protect users' personal information. This involves implementing techniques like differential privacy to ensure that individual data points cannot be reverse-engineered from the model outputs. Last but not least, *availability* ensures that the ML-enabled system remains accessible and operational, even under high load or during adverse conditions. This involves designing models and systems that can handle scalability challenges and remain responsive during peak usage times. By prioritizing resilience, designers and developers can build reliable and secure ML-enabled systems that can withstand disruptions and threats while maintaining their core functions and services. This altered understanding of the non-functional requirements within the resilience cluster highlights the unique challenges posed by integrating machine learning into complex systems.

Sustainability. Sustainability is chosen as a cluster to represent economic, social, and environmental aspects. It ensures that resources are used efficiently and effectively, reducing the environmental impact, *energy consumption* and financial *costs* associated with producing the model and making inferences. Additionally, sustainability is chosen to represent *fairness*, *ethics*, and *accountability*, as these aspects are critical for ensuring that the system benefits all stakeholders, including marginalized and vulnerable communities, and operates in a way that aligns with ethical principles and values. The rationale to classify social and ethical concerns under the *"Sustainability"* category comes from the increasing recognition that social sustainability is currently experiencing. For instance, the United Nations included the reduction of inequalities

among the 17 objectives for sustainable development.[12] Researchers have also been arguing that fairness and ethical concerns should be considered as a form of sustainability. McGuire et al. [47] advocated that social sustainability refers to multiple dimensions, including pro-social vs. anti-social affordances. On a similar note, various other researchers [8, 17, 36] argued to consider social and ethical aspects as sustainability properties. Overall, sustainability can be considered a guiding principle for designing and developing efficient, cost-effective but also equitable, ethical, and accountable systems. When comparing the definition of these non-functional requirements to those of non-ML-enabled systems, it is worth pointing out that some were altered to better fit the context of ML-enabled systems, while others represent new aspects specific to ML-enabled systems. In particular, *energy consumption* and *costs* assume a slightly different meaning when considered in this context. Indeed, the former refers not only to the energy used during the operation of the system but also to the significant resources required during the inference of machine learning models. This includes the computational power needed to make model predictions, which can have a considerable environmental impact. As for the latter, it includes both the financial costs associated with developing, deploying, and maintaining ML models, and the long-term operational costs. This encompasses costs related to data storage, processing power, and the need for specialized hardware. The other non-functional requirements are instead new. *Fairness* is a new requirement that addresses the need to ensure that ML-enabled systems operate without bias, providing equitable outcomes for all users. This includes implementing strategies to mitigate algorithmic bias and ensuring diverse and representative training data. *Ethics* refers to the principles guiding the design and deployment of ML-enabled systems to ensure they operate in a morally responsible manner. This includes considerations of the potential societal impacts of deploying these technologies. Finally, *accountability* involves ensuring that the actions and decisions made by ML-enabled systems can be traced and audited. This includes implementing mechanisms for tracking data provenance, model decisions, and providing transparency in how models operate and make decisions. By incorporating these redefined and new non-functional requirements under the sustainability cluster, designers and developers can create ML-enabled systems that are not only efficient and cost-effective but also socially and ethically responsible. This holistic approach ensures that the systems developed are sustainable in the broadest sense, benefiting the environment, economy, and society at large.

Usability. This cluster has non-functional requirements with unique definitions because of the black-box nature of ML-enabled systems - it is similar to the one defined by Habibullah et al. [23]. More particularly, this class was chosen as the cluster to represent *interpretability*, *imperceptibility*, *explainability*, and *transparency*: it is concerned with ensuring that a system is easy to use and understand for end users, reducing the cognitive load required to interpret the model's behavior, and making it more transparent and explainable. In this cluster, some non-functional requirements are redefined to better fit the context of ML-enabled systems, while others retain their original meaning, and new requirements are introduced. More particularly, *imperceptibility* retained its original meaning, referring to the system's ability to integrate seamlessly into the user's workflow without causing disruptions or requiring significant changes to existing processes. *Interpretability* is a new requirement that refers to the degree to which a human can understand the cause of a decision made by a model. This involves making

[12]The United Nations Goals for Sustainable Development: `https://sdgs.un.org/goals`.

the model's workings understandable to users, which is crucial given the complex and often opaque nature of machine learning algorithms. *Transparency* is a non-functional requirement that, while traditionally associated with characteristics of systems and processes in non-ML contexts, in ML-enabled systems focuses on providing clear insights into how the model functions, what data it uses, and the logic behind its decision-making process. This includes documenting the model's design, training data, and any preprocessing steps, ensuring users can trace and understand the model's operations. As for *explainability*, its meaning is altered to involve providing users with understandable and actionable explanations of the model's outputs. This goes beyond mere transparency by ensuring that users can comprehend and trust the decisions made by the system and understand the reasoning behind those decisions. By incorporating these specific non-functional requirements under the usability class, we address the unique challenges posed by the black-box nature of ML-enabled systems. This ensures that such systems are not only effective in their operations but also user-friendly, transparent, and trustworthy, ultimately enhancing their overall usability and acceptance among end users.

Researchers can use the taxonomy built in the context of our work to have a comprehensive mapping of the relevant non-functional requirements to optimize when developing ML-enabled systems, other than understanding the quality and efficiency aspects practitioners should focus on when releasing machine learning models.

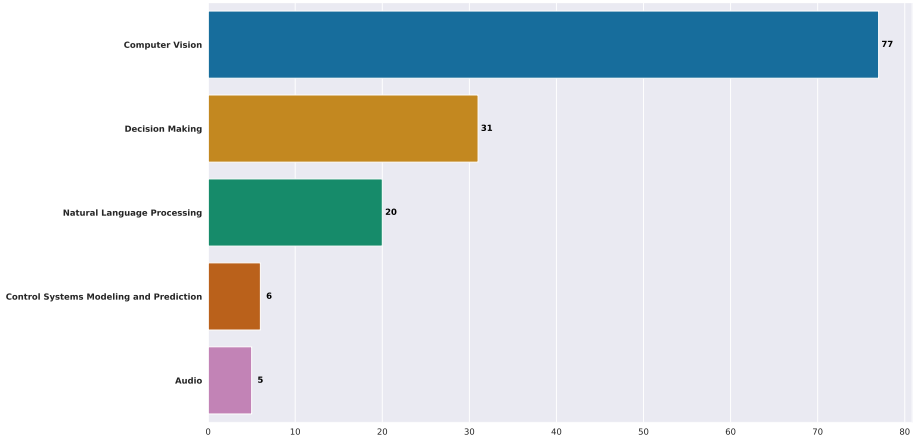

Figure 6: Distribution of NFRs across ML tasks in ML-enabled systems.

4.2.1. On the non-functional requirement domains

Figure 6 overviews the number of primary studies targeting each ML task extracted. ML tasks were defined following the conventions used by sources such as Hugging Face.[13] As shown, the most frequent are *'Computer Vision'* (77), *'Decision Making'* (31), and *'Natural Language Processing'* (20), while other, less targeted ML tasks pertain to emerging technologies, e.g., *'Control Systems Modeling and Prediction'* or *'Audio'*. These insights raise some key ML tasks that researchers have been analyzing in the recent past but also raise contexts where further research

[13] `https://huggingface.co/models`

might be worth focusing on. The much larger amount of primary studies targeting *'Computer Vision'* may indicate the critical nature of such ML tasks with respect to the management of non-functional requirements. This may be due to the nature of the inputs that the ML-enable systems should consider, i.e., images or videos, which may be the subject of multiple concerns such as accuracy, security, ethics, and privacy—this may potentially emphasize the need for novel methods to process images and/or deal with non-functional attributes in the field.

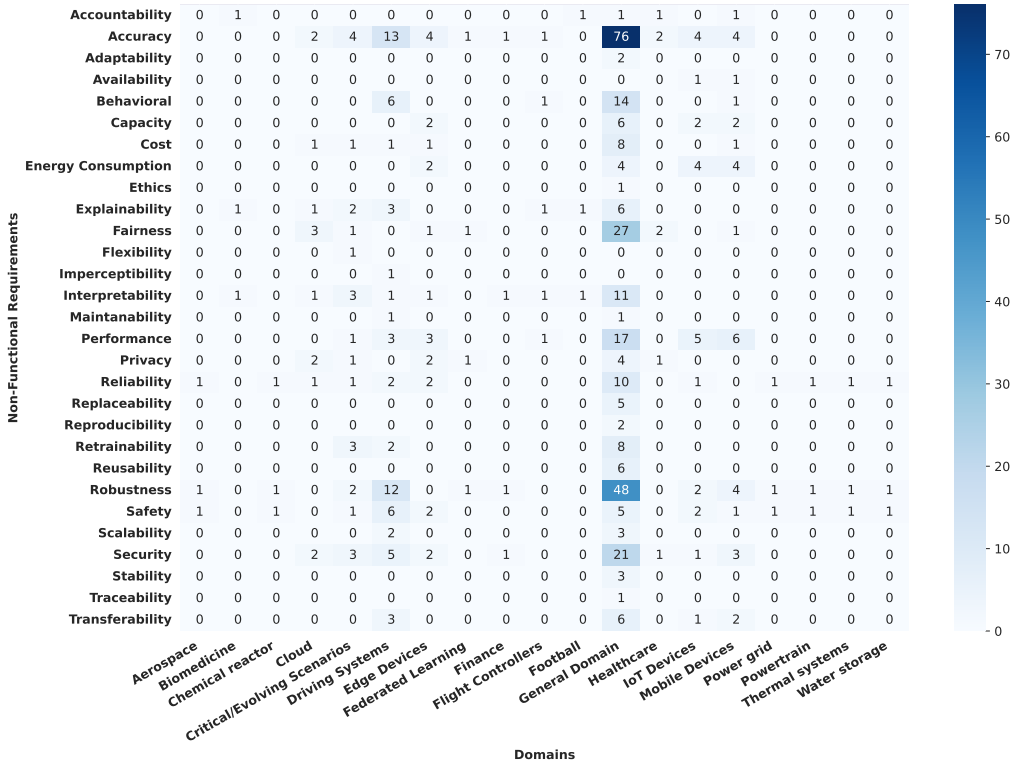

Figure 7: Heatmap of Non-Functional Requirements across domains in ML-enabled systems.

Providing an additional perspective, Figure 7 illustrates a heatmap that shows the relation between the specific non-functional requirements presented in Table 4 and the corresponding domains. Each cell in the heatmap contains a numerical value representing the frequency of a non-functional requirement (indexed by row i) considered within a particular domain (indexed by column j). Through this visualization, we could first observe that there exist several domains where the current knowledge seems to be limited: for example, the domains concerned with *'Finance'*, *'IoT'*, and *'Cloud'*, and *'Edge Devices'* were the least targeted ones, which possibly suggests that further research might consider the impact of non-functional requirements in these contexts. As expected, *accuracy* represents the property more frequently addressed in all the domains. Nonetheless, some non-functional requirements like *robustness*, *performance*, *fairness*, and *security* have been targeted much more than others, possibly indicating that researchers perceived these as essential properties to investigate. At the same time, Figure 7 could further highlight the non-functional requirements that were somehow neglected so far: for instance, all

the properties concerned with maintainability were found to be mostly unexplored independently from the domains considered.

In addition, Figure 7 helps highlight potential trade-offs among multiple non-functional requirements. In particular, the figure shows that in certain domains, multiple non-functional requirements are considered together more frequently. This suggests that addressing one non-functional requirement often necessitates compromises with others due to competing priorities and resource limitations. For instance, considering the *'Healthcare'* domain, we observed that accuracy coincides with fairness, privacy, and security, suggesting that these aspects are all critical in this domain. Consequently, prioritizing high accuracy may impact the resources and focus available to ensure fairness, privacy, and security, or vice versa. We believe that such a visualization provides readers with a deeper understanding of the non-functional requirements to consider when addressing specific domains and, more importantly, of the potential trade-offs to investigate in order to balance multiple non-functional requirements effectively.

☞ **Answer to $\mathbf{RQ}_1$.** As a result of $\mathbf{RQ}_1$, we could identify a total of 31 relevant non-functional attributes, classified according to six main categories. We also identified the most frequent ML tasks and ML domains where these non-functional attributes have been investigated. The resulting taxonomy maps the current knowledge on the matter and provides insights into the research areas that may be worth exploring in the future.

4.3. *$\mathbf{RQ}_2$ - What are the challenges for ML-enabled systems specifically relating to non-functional requirements?*

The second research question focuses on identifying the challenges faced by ML-enabled systems specifically relating to non-functional requirements. In the following, we summarize the data analysis process and results that address $\mathbf{RQ}_2$.

Data Analysis and Synthesis. Similarly to the previous research question, we addressed $\mathbf{RQ}_2$ by means of a three-step systematic classification exercise that aimed at (i) extracting the challenges for ML-enabled systems specifically relating to non-functional requirements, as reported in the primary studies, (ii) synthesizing them by grouping similar challenges, and (iii) refining the groups initially produced. In this respect, it is worth remarking that we deliberately did not include all challenges faced by ML-enabled systems but focused solely on those that directly impact the achievement, management, or optimization of specific NFRs. These challenges were derived exclusively from the primary studies selected as part of the systematic literature review, ensuring that all included challenges are traceable to evidence provided in the reviewed studies. Consequently, challenges that are not discussed in these papers, even if potentially relevant to NFRs, could not be included. The three steps were conducted as follows:

- *Data Extraction.* For each primary study, we extracted information about the nature of the challenges for ML-enabled systems specifically relating to NFRs, their underlying causes, potential impacts, and any specific examples or case studies provided. In this respect, it is worth remarking that only challenges explicitly linked to NFRs were included in our analysis. While the primary studies may discuss a broader range of issues, our focus remained on extracting and analyzing challenges directly associated with NFRs. For example, in studies addressing algorithmic discrimination, we analyzed challenges such as mitigating disparate impacts or ensuring equitable model performance—both of which are closely tied to the NFRs of fairness and accuracy. Conversely, challenges mentioned in the papers

Table 5: Challenges Related to Non-Functional Requirements in ML-Enabled Systems.

Cluster	Challenge	References
Efficiency	E.1 - Dealing with High Latency	[S8, S12, S21, S57, S94, S101]
	E.2 - Dealing with Space	[S4, S5, S8, S11, S81, S99, S101, S125]
	E.3 - Managing Simulation for Inference	[S4, S5, S81, S125]
Maintainability	M.1 - Increasing Model Reproducibility	[S49]
	M.2 - Increasing Model Decomposition and Reuse	[S16, S49, S54, S90, S91, S94]
Resiliency	R.1 - Resilience to Adversarial Attacks	[S28, S29, S56, S73, S98, S108]
	R.2 - Exploiting the Sensitivity of Adversarial Attacks	[S17, S28, S29, S85, S110, S111, S116]
	R.3 - Security Verification of Pre-Trained Models	[S25, S46, S89, S106, S112]
	R.4 - Resilience to Intellectual Property Theft	[S23, S106]
	R.5 - Optimal Post-Deployment Simulation	[S41, S42, S68, S82, S84, S92, S93]
	R.6 - Preserving Privacy in Machine Learning-Enabled Systems	[S3, S34]
Sustainability	S.1 - Dealing with Algorithmic Discrimination	[S2, S20, S22, S37, S38, S102, S103]
	S.2 - Model Accountability	[S96, S119, S122]
	S.3 - Fairness Analytics	[S1, S34, S48, S64, S69, S76, S78, S83, S86, S104, S119]
	S.4 - Improving Sustainability Benchmarks	[S97, S122]
	S.5 - Reducing Energy Cost	[S5, S8, S12, S26, S30, S51, S52, S81, S99, S100, S130]
	S.6 - Increasing the Practitioner's Awareness of Sustainability	[S5, S51, S125]
Usability	Usability	[S8, S9, S31, S32, S60, S70, S72, S87, S113, S120, S124]
Cross-cutting challenges	C.1 - Dealing with Internal Errors	[S5, S8, S27, S62, S63, S88]
	C.2 - Diagnosing the Internal Behavior of Models	[S6, S10, S24, S33, S40, S59, S66, S67, S74, S75, S75, S114]
	C.3 - Context-based Trade-off Identification	[S34, S37, S49, S49, S50, S126]
	C.4 - Prioritizing and Balancing Non-Functional Requirements	[S1, S4, S5, S11, S32, S37, S38, S51, S64, S86, S100, S104]
	C.5 - Integration and Interaction of ML Models	[S61, S65, S70, S95, S118]
	C.6 - Improving Practitioners' Knowledge	[S69, S105, S129, S130]
	C.7 - Software Analytics for Non-Functional Requirement Assessment	[S12, S13, S26, S120, S121, S121]
	C.8 - Improving the Generalizability of Existing Automated Approaches	[S19, S36, S50, S53]

but unrelated to NFRs, such as purely technical optimizations without a clear NFR connection, were excluded. Similarly, we excluded generic ML system challenges that do not intersect with NFRs. For instance, issues like hyperparameter tuning or general convergence optimization were excluded unless the primary studies explicitly linked these issues to their impact on a specific NFR.

- *Synthesis of Challenges.* We synthesized the extracted data by grouping similar challenges together and identifying common themes. This process involved analyzing the context in which each challenge was presented and understanding the broader implications of the challenges described for ML-enabled systems.

- *Validation and Refinement.* The synthesized challenges were initially extracted by the first author of the paper. Subsequently, these challenges underwent a review and validation process through discussions with the second author. These discussions aimed to ensure consistency and accuracy in the categorization and description of the challenges. During this stage, only minor modifications were made to the names and descriptions to enhance clarity and coherence.

Addressing RQ_2. Our work identified 26 software engineering challenges for our research community summarized in Table 5. First and foremost, it is worth pointing out that most of the challenges identified pertained to the use/integration of neural networks within more complex

software systems. This was somehow expected, as deep learning has become the most widely employed form of machine learning to empower the capabilities of traditional software systems [21]. In the second place, accuracy represents a cross-cutting concern, i.e., all the challenges identified have implications for the accuracy of ML-enabled systems. For this reason, we preferred to discuss accuracy while presenting the challenges connected to the other classes of NFRs identified in $\mathbf{RQ}_1$. The list of challenges is presented below.

Efficiency (E): In terms of the overall performance and effectiveness of ML-enabled systems, we could elicit three major challenges, which are connected to internal errors, high latency, and memory issues. Specifically:

- **Challenge E.1 - Dealing with High Latency [S8, S12, S21, S57, S94, S101]**. This challenge is connected to *performance* requirements, hence affecting the efficiency of ML-enabled systems.

  Nature of the Challenge. Dealing with high latency in ML-enabled systems, particularly in neural networks, is a significant challenge. This latency can arise from various design choices and optimization strategies that aim to improve accuracy but may inadvertently increase processing time.

  Possible Underlying Causes. Multiple design choices related to *hyper-parameter tuning*, *model optimization*, and *model architecture* can affect the overall efficiency of neural networks. These decisions, while sometimes leading to increased accuracy, can result in higher latency. Additionally, the computational efficiency of DNN systems is very sensitive to even slightly different inputs. Changes to the inputs may result in significantly higher computational demand, deteriorating the overall models' efficiency.

  Potential Impacts: High latency can severely impact the usability and performance of ML-enabled systems, leading to slower response times and decreased user satisfaction. It can also affect the real-time processing capabilities of the system, making it unsuitable for applications requiring quick decision-making.

  Target Research: Researchers in the fields of *software analytics*, *software architecture*, and *software quality and optimization* can help address this challenge by developing analytical and code quality tools aimed at finding the right compromise between accuracy and efficiency. This includes conducting preliminary studies to identify the best configurations for specific use cases and optimizing the computational efficiency of neural networks to handle varying input conditions effectively.

- **Challenge E.2 - Dealing with Space [S4, S5, S8, S11, S81, S99, S101, S125]**. This challenge affects both *performance* and *capacity* of ML-enabled systems.

  Nature of the Challenge. A critical challenge identified is the space required by machine learning models, particularly neural network-based solutions. These models can be very large, leading to significant memory issues.

  Possible Underlying Causes. The large size of neural network models is particularly relevant in contexts like edge computing and IoT. Deploying these large models on such devices often results in memory constraints that can affect overall system performance.

  Potential Impacts: Memory constraints caused by large models can lead to degraded performance, reduced responsiveness, and the inability to deploy models on resource-limited devices such as those used in edge computing and IoT domains.

Target Research: Designers should apply policies and strategies to build reduced models before deploying them on devices to mitigate memory issues and improve overall performance. This involves leveraging techniques in the emerging field of *tiny machine learning* [38], which combines hardware, algorithms, and software to support on-device sensor data analytics at low power. While some initial efforts have been made in the artificial intelligence research community (e.g., [18, 63]), there is a notable lack of knowledge and research from the software engineering perspective. Further research is needed to develop efficient and scalable solutions to this challenge, ensuring that ML-enabled systems can operate effectively in resource-constrained domains.

- **Challenge E.3 - Managing Simulation for Inference [S4, S5, S81, S125].** This challenge impacts *performance* requirements of ML-enabled systems.

  Nature of the Challenge. The selection of devices for deploying an ML model can significantly impact its *performance*, particularly for edge models used in ML inference, where *latency* and *power consumption* are critical factors. Traditional methods and existing benchmarks often fail to provide a comprehensive and reproducible evaluation of these non-functional requirements across various edge platforms and ML models.

  Possible Underlying Causes. This challenge is further complicated by the diversity of edge hardware and associated software toolkits, which necessitate benchmarks that can handle different configurations and optimizations. The lack of standardized benchmarks that accommodate the wide range of model and platform combinations makes it difficult to evaluate performance consistently.

  Potential Impacts: These challenges affect ML-enabled systems by making it difficult to evaluate a wide range of model and platform combinations. This complicates developers' ability to make informed decisions for specific deployment scenarios, potentially leading to suboptimal performance, increased latency, or excessive power consumption, especially on edge or IoT devices with hardware limitations.

  Target Research: Researchers in the fields of *software analysis* and *software quality* should develop tools that simulate model efficiency before deployment in physical domains. This is especially crucial for edge or IoT devices, where balancing competing objectives, such as detection accuracy, frame processing rate, and energy consumption, is particularly challenging. By focusing on creating robust simulation tools, researchers can help developers better evaluate and optimize ML models for specific hardware configurations, ultimately improving the deployment and performance of ML-enabled systems in resource-constrained domains.

Maintainability (M): When it turns to the ability of ML-enabled systems to be adapted and evolved over time, we could elicit two main challenges connected to model reproducibility and model decomposition and reuse.

- **Challenge M.1 - Increasing Model Reproducibility [S49].** It revolves around the clusters of *"Accuracy"* and *"Maintainability"*, with *accuracy* and *reproducibility* impacting ML-enabled systems:

  Nature of the Challenge. ML-enabled systems based on deep learning models are known to be complex and difficult to reproduce accurately. Ensuring that these models can be reliably reproduced is a significant challenge in the field.

Possible Underlying Causes. Reproducing deep learning models can be challenging because of randomness, which can affect the behavior of the algorithms used. Additionally, hardware non-determinism, such as that present in graphics processing units (GPUs), can further complicate the reproducibility of deep learning models.

Potential Impacts: The inability to accurately reproduce deep learning models can lead to inconsistent results, making it difficult to validate and verify model performance. This inconsistency undermines the trust in model outputs and hinders the ability to build upon previous research or deploy models in production environments with confidence.

Target Research: Researchers in the filed of *software quality* and *software testing* should focus on designing mechanisms that support the *verification* of deep learning models. This can be achieved, for instance, by injecting elements of randomness and non-determinism to observe how the system behaves under those conditions. By understanding and controlling for these variables, researchers can improve the reproducibility of deep learning models, ensuring that they produce consistent and reliable results across different runs and hardware configurations. This will enhance the reliability and trustworthiness of ML-enabled systems in practical applications.

- **Challenge M.2 - Increasing Model Decomposition and Reuse [S16, S49, S54, S90, S91, S94].** This challenge affects non-functional requirements such as *replaceability*, *adaptability*, and *reproducibility* of ML-enabled systems:

  Nature of the Challenge. When building and improving ML-enabled systems based on deep learning models, there is the need to reuse parts of previously constructed models, which is essential for maintaining and evolving models efficiently.

  Possible Underlying Causes. This challenge arises when attempting to replace potentially defective parts with others or when selectively reusing components of a model. Identifying the parts responsible for each output class or module in the models can be difficult due to the intricate and often opaque nature of deep learning architectures. Additionally, the properties of randomness and non-determinism in both software and hardware components further complicate this process.

  Potential Impacts: The inability to effectively decompose and reuse model components can lead to inefficiencies in model development and maintenance. It can also result in increased costs and longer development times, as engineers may need to rebuild models from scratch rather than reusing existing components. Furthermore, it hampers the flexibility to quickly adapt models to new tasks or improve specific functionalities.

  Target Research: Researchers should focus on developing methods for component *verification* and identifying reusable components through *program slicing* and *refactoring*. These approaches can help ML engineers accurately extract parts of models for reuse and replace defective or outdated components efficiently. By improving the understanding of the sources of variability and non-determinism, and enhancing reproducibility, it becomes easier to decompose models into reusable units. This not only increases the efficiency of model development and maintenance but also ensures that ML-enabled systems can be more easily and reliably updated and adapted to meet new requirements.

Q Resiliency (R): Improving machine learning models' universal robustness and security presents key issues [S44]. This was the class of non-functional requirements where we identified more

challenges (7); these were connected to various aspects such as adversarial attacks, theft of model and intellectual property, attack and defense of models, repairing internal models, post-deploy issues, preserving privacy, and vulnerability transferability.

- **Challenge R.1 - Resilience to Adversarial Attacks [S28, S29, S56, S73, S98, S108].** This challenge impacts *"Accuracy"* and *"Resiliency"*, particularly concerning *accuracy*, *robustness*, and *security* for ML-enabled systems:

  Nature of the Challenge. Machine learning models, particularly deep neural networks, are vulnerable to adversarial attacks. These attacks manipulate input data to cause the neural network to produce incorrect outputs, significantly impacting the model's *accuracy* and overall *robustness*.

  Possible Underlying Causes. Different types of adversarial attacks can be targeted or untargeted and can be carried out through various techniques such as *adversarial examples* [66], *adversarial perturbation* [7], *poisoned data* [28], and *backdoor samples* [22]. These attacks exploit vulnerabilities in the model, leading to incorrect predictions and reducing the model's reliability.

  Potential Impacts: Adversarial attacks can severely degrade the performance and reliability of ML-enabled systems. They can cause models to make erroneous decisions, which in critical applications such as autonomous driving, healthcare, or security systems, could lead to catastrophic consequences. This challenge emphasizes the need for robust defenses and improved resilience to maintain system integrity under attack.

  Target Research: Researchers in the field of *software testing* and *software security* should focus on developing instruments that can design perturbations to generate adversarial examples for different machine learning models, especially in computer vision problems. These tools would help practitioners understand how models can be affected by vulnerabilities and enhance methods to defend against adversarial attacks. Additionally, the focus should be on the *verification and validation* of ML-enabled systems. While the artificial intelligence community has targeted the *security* perspective by developing new algorithms and techniques [37, 70], complementary software engineering research could propose novel verification and validation approaches. These approaches would improve the strategies and policies used by practitioners to test for security, ultimately enhancing system robustness and reliability against adversarial attacks.

- **Challenge R.2 - Exploiting the Sensitivity of Adversarial Attacks [S17, S28, S29, S85, S110, S111, S116].** This challenge focuses on non-functional aspects related to *security*, *robustness*, and *availability* of ML-enabled systems:

  Nature of the Challenge. A complementary perspective to dealing with adversarial attacks is understanding the sensitivity of certain adversarial attacks to input distortions and how this sensitivity can be exploited to enhance the *security*, *robustness*, and *availability* of machine learning models.

  Possible Underlying Causes. In computer vision applications, attacks made through non-additive noise or geometric morphing can cause significant security concerns. These attacks are sensitive to input distortions, which means that automatic correction instruments could potentially distort the malicious inputs, thereby mitigating their effects on the machine learning model.

Potential Impacts: The ability to exploit the sensitivity of adversarial attacks to distortions can enhance the security and robustness of ML models, making them more resistant to adversarial manipulation. This can reduce the risk of incorrect model outputs due to malicious attacks and improve the reliability and availability of ML-enabled systems.

Target Research: Researchers in the broad areas of *software testing* and *software security* should focus on developing techniques that can handle perturbations not only occurring naturally in the physical environment but also those generated maliciously by adversarial attacks. These perturbations are typically stealthy and undetectable, reducing the *flexibility* and *behavioral robustness* of deep neural networks. Characterizing these perturbations is a crucial first step for researchers in *empirical software engineering* and *automatic program repair*. By understanding the nature of these adversarial perturbations, researchers can develop methods to automatically correct or distort the malicious inputs, thereby enhancing the overall security and robustness of machine learning models against adversarial attacks.

- **Challenge R.3 - Security Verification of Pre-Trained Models [S25, S46, S89, S106, S112].** This challenge revolves around the non-functional related to *security* and *privacy* of ML-enabled systems:

  Nature of the Challenge. Pre-trained models can be highly beneficial as they reduce the computational burden of training complex deep learning models through transferability. However, they are susceptible to security vulnerabilities due to dataset displacement and the potential for malicious code or exploitation.

  Possible Underlying Causes. The displacement of the dataset used for pre-training and the potential inclusion of malicious code during pre-training can introduce security risks in pre-trained models. These vulnerabilities can compromise the overall *security* of the ML-enabled system.

  Potential Impacts: Using pre-trained models without proper security verification can lead to the deployment of models that are vulnerable to attacks, potentially resulting in compromised system integrity and performance. This can have severe consequences, especially in critical applications where security is paramount.

  Target Research: Research in the area mainly targets *software analytics*, *software quality*, and *verification and validation* researchers. At first, research should focus on developing software analytics instruments that can provide practitioners with security insights and best practices for selecting the most appropriate pre-trained models to avoid security concerns. Additionally, the definition of design practices for developing and retraining security-aware fine-tuned models is crucial. Existing research highlights the need for tools and methodologies that can assess the security risks of pre-trained models and ensure their reliability. Software engineering researchers may contribute significantly to addressing these challenges by developing and implementing security verification processes for pre-trained models, thus enhancing their safety and robustness in practical applications.

- **Challenge R.4 - Resilience to Intellectual Property Theft [S23, S106].** This challenge mostly affects *privacy*, *robustness* and *safety* of ML-enabled systems:

  Nature of the Challenge. ML-enabled systems based on neural networks are vulnerable to data breaches and unauthorized access to sensitive data due to training attacks.

These attacks can involve malicious manipulation of training data [68], undermining the *robustness*, *safety*, and *privacy* of the system.

Possible Underlying Causes. Training attacks, which target the data used to train machine learning models, can lead to unauthorized access and intellectual property theft. These attacks exploit weaknesses in the security protocols of ML-enabled systems, particularly during the training phase.

Potential Impacts: Such vulnerabilities can result in the theft of proprietary algorithms and sensitive data, causing significant financial and reputational damage. It can also lead to the unauthorized use or modification of models, compromising their integrity and effectiveness.

Target Research: According to existing research, there is limited knowledge on securing machine learning models against these types of attacks. Greater attention should be paid to strengthening security protocols. While this issue primarily concerns researchers in the field of networks and security, the software engineering research community can also contribute by defining security best practices that protect models from intellectual property theft. This is particularly crucial for edge devices and IoT systems, where the risk of unauthorized access is higher [75]. By developing robust security measures and protocols, software engineers can help ensure that machine learning models are safeguarded against intellectual property theft, thereby enhancing the overall resilience and trustworthiness of ML-enabled systems.

- **Challenge R.5 - Optimal Post-Deployment Simulation [S41, S42, S68, S82, S84, S92, S93].** This challenge impacts both *robustness* and *safety* of ML-enabled systems:

  Nature of the Challenge. While some threats to resiliency might be managed during development, additional challenges arise at the deployment stage. Machine learning models might not work as expected when deployed, which can deteriorate the *robustness*, *safety*, and *reliability* of the system.

  Possible Underlying Causes. The primary issue is the difficulty in understanding how the model will perform in real-world scenarios. Factors such as diverse environmental conditions, varied input data, and unforeseen operational challenges can impact the model's performance post-deployment. This challenge has been noted in specific contexts, such as audio applications, but it can affect any kind of ML solution.

  Potential Impacts: Unexpected performance issues post-deployment can lead to failures in critical applications, potentially causing significant harm or financial loss. It can also reduce user trust and system reliability, making it crucial to address these challenges to ensure the system functions correctly in real-world conditions.

  Target Research: Research has identified post-deployment simulations as a complementary instrument to traditional software testing [58]. These simulations involve the definition of agent-based models that can (i) simulate the environment where the ML-enabled system would operate and (ii) verify the system against a wide range of simulated inputs. Such simulations can help in understanding and predicting the model's behavior in real-case scenarios, thus enhancing its *robustness*, *safety*, and *reliability*. Researchers in the fields of *verification and validation* and *machine learning for software engineering* should focus on developing and refining these simulation techniques to address post-deployment challenges effectively. By doing so, they can contribute to creating more

resilient and reliable ML-enabled systems that perform consistently in real-world applications.

- **Challenge R.6 - Preserving Privacy in Machine Learning-Enabled Systems [S3, S34].** Non-functional requirements related to *security* and *privacy* for ML-enabled systems are impacted by this challenge:

  Nature of the Challenge. The challenge concerns the development of privacy-preserving deep neural networks and machine learning systems. This involves creating algorithms that can operate securely on distributed data without compromising sensitive information about the data subjects, while allowing users to delete their data at any time.

  Possible Underlying Causes. Ensuring privacy in ML-enabled systems requires strategies that protect private data during training and inference processes. The challenge lies in developing methods that allow models to learn from data without directly accessing or storing sensitive information, which includes data encryption, anonymization, federated learning, and differential privacy techniques.

  Potential Impacts: Failing to preserve privacy can lead to data breaches and unauthorized access to sensitive information, undermining user trust and violating data protection regulations. Privacy-preserving methods are crucial for maintaining the *privacy*, *security*, and protection of personal data while ensuring the *accuracy* of the models.

  Target Research: This challenge targets multiple software engineering fields, from *software quality* to *software architecture*. Researchers need to focus on deepening the current knowledge of data encryption, anonymization techniques, federated learning, and differential privacy. These strategies are essential for training models without sacrificing *accuracy* while ensuring the *privacy*, *security*, and protection of private data. By advancing these techniques, the software engineering community can contribute to developing robust privacy-preserving ML-enabled systems that maintain user trust and comply with data protection standards.

Sustainability (S): As for the challenges in this category, we could observe that most of the available literature focused on ethics and fairness, somehow neglecting other perspectives of sustainability. More particularly, we identified six main challenges connected to algorithmic discrimination, model accountability, fairness metrics, low-quality datasets, energy cost, energy and performance aware trade-offs, and generalizability of existing solutions.

- **Challenge S.1 - Dealing with Algorithmic Discrimination [S2, S20, S22, S37, S38, S102, S103].** This challenge affects *"Accuracy"* and *"Sustainability"* and in particular *accuracy* and *fairness* of ML-enabled systems:

  Nature of the Challenge. Improving *performance*, *accuracy*, and *fairness* simultaneously remains a significant challenge for researchers. Existing methods for generating discriminatory instances often fail to produce realistic samples. The primary concern is studying and mitigating the impact of disparate results, offensive labeling, and uneven algorithmic error rates in data-driven applications.

  Possible Underlying Causes. The root causes of algorithmic discrimination include biases present in training data, model design, and deployment contexts. Discrimination

can occur during the pre-processing phase, where data is prepared for model training, leading to biased outcomes that favor certain groups over others.

Potential Impacts: Algorithmic discrimination can lead to unfair and biased outcomes, reducing the trustworthiness and ethical integrity of ML systems. This can have severe societal impacts, particularly in applications like hiring, lending, and law enforcement, where biased algorithms can perpetuate and amplify existing inequalities.

Target Research: The primary studies highlighted the importance of addressing algorithmic discrimination, especially during the pre-processing phase, to improve the trade-off between *fairness* and *performance* in ML software. The software engineering research community has made some strides in this area (e.g., [S114, S119]), but further research is necessary. Focus areas include *software analytics* and *verification and validation*, which can provide tools and methodologies to detect and mitigate biases, ensuring more equitable and fair ML-enabled systems. By advancing these research areas, the community can develop robust methods to address algorithmic discrimination effectively, contributing to the ethical deployment of ML technologies.

- **Challenge S.2 - Model Accountability [S96, S119, S122].** This challenge mostly affects *"Accuracy"* and *"Sustainability"*, particularly touching non-functional requirements concerning *accuracy*, *fairness*, *accountability*, and *ethics* of ML-enabled systems:

  Nature of the Challenge. The term *'accountability'* refers to the model's ability to provide clear explanations of its decisions, its transparency in how it has been trained and makes decisions, and its ability to allow users to provide feedback or challenge its decisions. Ensuring model accountability involves making the decision-making process understandable and transparent, which is key for ethics and fairness.

  Possible Underlying Causes. The underlying causes of the lack of model accountability include complex and opaque model architectures, insufficient documentation of training processes, and the absence of mechanisms for users to query or challenge decisions. These issues are compounded by the rapid development and deployment cycles of ML systems, which often prioritize performance over transparency.

  Potential Impacts: Without accountability, ML-enabled systems can make decisions that are difficult to understand or explain, leading to mistrust and potential misuse. This can have significant ethical implications, especially in critical areas such as healthcare, criminal justice, and finance, where decisions can significantly impact individuals' lives.

  Target Research: The primary studies considered in our work emphasized the importance of focusing more effort on model accountability to ensure the ethical and fair deployment of ML systems. The research should prioritize the development of *verification and validation* instruments that allow practitioners to verify how accountable their systems actually are. For instance, decision-making frameworks can serve as ideal tools to assess the level of *accountability* of machine learning models. By focusing on these verification and validation processes, researchers and practitioners can ensure that models are not only accurate but also transparent and responsive to user feedback, thus fostering trust and ethical integrity in ML-enabled systems.

- **Challenge S.3 - Fairness Analytics [S1, S34, S48, S64, S69, S76, S78, S83, S86, S104, S119].** This challenge impacts on both *"Accuracy"* and *"Sustainability"*, and in partic-

ular non-functional requirements concerning *accuracy*, *fairness*, and *accountability* of ML-enabled systems:

Nature of the Challenge. Fairness analytics involves defining metric toolkits that support practitioners in diagnosing fairness during the development of ML-enabled systems. The challenge lies in creating tools that help identify and measure fairness-related issues early in the development process. However, fairness should not be considered a non-functional requirement in isolation but rather as an integral part of trade-off analyses.

Possible Underlying Causes. The primary cause of this challenge is the inherent complexity of balancing fairness with other competing requirements such as accuracy, privacy, and accountability. Existing tools and metrics often lack the capability to provide a comprehensive analysis that considers these trade-offs simultaneously. The dynamic and context-dependent nature of fairness further complicates the development of universal metrics and toolkits.

Potential Impacts: If fairness is not adequately addressed, ML-enabled systems can perpetuate biases and inequalities, leading to unfair and discriminatory outcomes. This can erode trust in the system, reduce its acceptance and reliability, and cause harm to individuals and communities.

Target Research: The literature calls for novel metrics that provide insights into the trade-offs between *fairness* and other critical non-functional requirements such as *accuracy*, *privacy*, and *accountability*, as well as combinations of these factors. The research highlights the need for diagnosis and multi-objective optimization of data-driven applications across various stages of development, from requirements engineering to verification and validation. To address these challenges, researchers should focus on developing comprehensive metric toolkits that can diagnose fairness issues and provide multi-objective optimization solutions. These toolkits should be integrated into the development workflow to ensure that fairness is considered alongside other non-functional requirements from the outset. By doing so, practitioners can make informed decisions that balance fairness with other critical system attributes, leading to more equitable and trustworthy ML-enabled systems.

- **Challenge S.4 - Improving Sustainability Benchmarks [S97, S122].** Such a challenge may impact various aspects, including *accuracy*, *fairness*, and *accountability* of ML-enabled systems:

  Nature of the Challenge. Improving sustainability benchmarks involves creating high-quality datasets that assist in developing sustainable ML applications. The challenge is to develop datasets that enable comprehensive cross-domain and cross-sectional analysis while maintaining high standards of fairness and accountability.

  Possible Underlying Causes. The primary cause of this challenge is the prevalence of low-quality datasets that can exacerbate issues related to discrimination, fairness, and accountability. Low-quality datasets often fail to represent diverse populations adequately, leading to biased training and evaluation of machine learning models. Additionally, the need for specialized datasets in certain application domains, such as image processing, further complicates the creation of comprehensive benchmarks.

  Potential Impacts: Low-quality benchmark datasets can lead to the development of ML systems that are biased and unreliable. This can negatively impact the fairness and

accountability of these systems, leading to discriminatory outcomes and reducing trust in the technology. Furthermore, high-quality benchmark datasets may be essential to ensure that ML models are trained and evaluated in a way that is fair, accurate, and representative of diverse populations.

Target Research: Current research has highlighted the need for improved benchmark datasets to create sustainable ML applications. Low-quality datasets are more likely to discriminate against individuals in the dataset and worsen fairness and accountability attributes. Therefore, there is a pressing need for high-quality datasets that enable automated pose, illumination, and expression (PIE) analysis in image processing applications. Additionally, datasets that present large and diverse information on minorities are crucial for more effective training of ML systems. Researchers should focus on developing these high-quality datasets by ensuring they cover diverse populations and various domains comprehensively. This involves collaborative efforts across different fields and leveraging advanced data collection and annotation techniques to create benchmarks that truly reflect the complexities and nuances of real-world data. By improving the quality and diversity of benchmark datasets, the research community can enhance the sustainability, fairness, and accountability of ML-enabled systems, ultimately leading to more robust and trustworthy applications.

- **Challenge S.5 - Reducing Energy Cost [S5, S8, S12, S26, S30, S51, S52, S81, S99, S100, S130].** As the name of the challenge suggests, it impacts both *energy consumption* and *cost* of ML-enabled systems:

  Nature of the Challenge. Reducing the energy and computational costs of deep neural networks is a critical challenge. These systems often require substantial energy and financial resources, leading to increased CO2 emissions and memory consumption during both the training and post-release phases.

  Possible Underlying Causes. Deep neural networks are inherently resource-intensive due to their complex architectures and large training datasets. The significant computational power needed for training these models results in high energy consumption. Additionally, post-deployment phases, especially on edge and IoT devices, exacerbate energy consumption issues due to the need for continuous processing and inference.

  Potential Impacts: High energy and computational costs can limit the scalability and deployment of ML-enabled systems, particularly in edge and IoT environments where resources are constrained. This not only increases operational costs but also has a significant environmental impact due to higher CO2 emissions. Moreover, it can lead to increased memory consumption, affecting their efficiency and sustainability.

  Target Research: This challenge may involve researchers in the board fields of *empirical software engineering*, *software quality and architecture*, and *machine learning engineering*. The former may be involved in assessing the impact of different AI containerization strategies on energy consumption and memory utilization. By empirically evaluating these strategies, it is possible to identify methods that minimize energy usage while maintaining performance. Researchers in software quality and architecture may focus on balancing the trade-offs between training accuracy and post-deployment energy consumption. This includes optimizing model architectures and developing energy-efficient training techniques. Finally, machine learning engineering researchers may be specifically looking at the rising field of tiny machine learning, which combines hardware,

algorithms, and software to support on-device sensor data analytics at low power. This field aims to create efficient ML models that can operate on resource-constrained devices without compromising performance.

- **Challenge S.6 - Increasing the Practitioner's Awareness of Sustainability [S5, S51, S125].** This challenge affects non-functional requirements concerning *energy consumption* and *cost* of ML-enabled systems:

  Nature of the Challenge. Increasing practitioners' awareness regarding sustainability concerns is a critical challenge identified in our literature review. This issue is especially relevant to energy consumption, which is a significant concern in today's environmentally conscious world. The challenge lies in making developers aware of the trade-offs between performance and sustainability.

  Possible Underlying Causes. Many practitioners may prioritize performance and accuracy over sustainability due to a lack of awareness or tools that highlight the environmental impact of their decisions. Additionally, existing development practices and tools may not emphasize the importance of sustainability, leading to its neglect during the software development lifecycle.

  Potential Impacts: If practitioners remain unaware of sustainability concerns, ML-enabled systems may continue to be developed with high energy consumption and resource usage, negatively impacting the environment. This can lead to inefficiencies and higher operational costs, reducing the overall sustainability of ML projects.

  Target Research: Addressing this challenge would require the effort of multiple communities. *Empirical software engineering* researchers may work toward understanding practitioners' needs and practices. By tailoring automated approaches to fit these needs, researchers may develop tools and methods that highlight the sustainability impacts of various development decisions. This could include developing frameworks that provide feedback on energy consumption and resource usage during the development process. At the same time, *software quality*, *software analytics*, and *machine learning engineering* researchers may target the design of automated approaches able to improve the scalability of models without compromising their overall accuracy. In this respect, approaches such as quantization [31] and knowledge distillation [62] can help reduce the energy footprint of ML models, making them more sustainable. Also, researchers in *software engineering education* may play a role in helping to create a common understanding of sustainability practices in software engineering.

Usability (U): Usability is the non-functional requirement with fewer challenges for ML-enabled systems. Only a few works targeted this matter, suggesting that further research might be worthwhile. In this context, non-functional requirements such as *explainability* and *imperceptibility*, along with the need for visualizing the outcomes of both shallow and deep learning models, pose significant challenges when it comes to assessing quality assurance throughout the software lifecycle. For instance, the lack of *interpretability* increases the effort required to estimate the soundness of ML-enabled systems. Current research neglected this research angle, especially when considering models operating in critical and evolving scenarios. As such, our findings suggest the urgent need for further research on usability concerns [S8, S9, S31, S32, S60, S70, S72, S87, S113, S120, S124]. Ensuring usability of AI algorithms

and their decisions can be a challenge for users and regulators, bringing distrust of users and stakeholders [S87, S127–S129]. This further suggests the investigation of domain-specific approaches and methods.

🔍 **Cross-cutting challenges (C):** Besides the challenges for ML-enabled systems specifically relating to individual classes of non-functional requirements, we also identified a set of cross-cutting challenges associated with the management and assessment of these requirements. It is worth remarking that the list of associated NFRs for each cross-cutting challenge reflects the specific NFRs that were most frequently and explicitly linked to the challenge in the primary studies included in our systematic literature review. This means that the associations presented are not exhaustive and do not exclude the possibility that a given challenge may also apply to other NFRs. Instead, the associations highlight the NFRs that were directly discussed in the context of each challenge within the reviewed papers.

- **Challenge C.1 - Dealing with Internal Errors [S5, S8, S27, S62, S63, S88].** This challenge mostly affects *"Efficiency"*, *"Sustainability"*, and *"Resiliency"*, and in particular non-functional requirements concerning *robustness*, *accuracy*, *performance*, *energy consumption* and *cost* of ML-enabled systems:

  Nature of the Challenge. Machine learning models and, in particular, neural networks can produce erroneous outputs due to internal errors caused by *incorrect parameters*, *incorrect weight values*, *uncovered root causes*, and *incorrect manual labeling*.

  Possible Underlying Causes. Internal errors can stem from several factors, such as misconfigured parameters, incorrect weight values assigned during the training process, overlooked root causes in the data or model structure, and errors introduced during manual labeling of training data.

  Potential Impacts: These internal errors may significantly impact the performance of ML-enabled systems, leading to reduced accuracy, slower inference times, and increased energy consumption. For example, incorrect weight values or parameters can cause the network to converge slowly or get stuck in suboptimal solutions, resulting in poor performance.

  Target Research: Dealing with internal errors has two main aspects: tracking errors in deep neural network models and addressing them. As a consequence, researchers in the fields of *bug localization*, *software analytics*, and *program repair* may help address the elicited challenges. For instance, *bug localization* researchers may work toward the development and refinement of novel techniques to effectively identify and localize bugs within neural network models by leveraging techniques such as fault injection, automated debugging, or static analysis. Research in *software analytics* may revolve around the definition of advanced monitoring frameworks that track a wide range of performance metrics, including accuracy, latency, throughput, and resource utilization. Finally, *program repair* researchers may provide automated instruments that suggest and implement fixes by leveraging AI algorithms.

- **Challenge C.2 - Diagnosing the Internal Behavior of Models [S6, S10, S24, S33, S40, S59, S66, S67, S74, S75, S75, S114].** This challenge affects multiple clusters, including *"Accuracy"*, *"Resiliency"*, and *"Usability"*. It particularly impacts *accuracy*, *robustness*, *safety*, and *transparency* of ML-enabled systems:

Nature of the Challenge. Understanding the root cause of anomalous behaviors of deep neural networks and how to fix them represents a critical aspect to further elaborate.

Possible Underlying Causes. Evaluating the *reliability* implications of small sets of weights assigned to the network can support practitioners in better understanding how they work. Inconsistent weight assignments, faulty training data, or model architecture flaws can lead to unpredictable or erroneous model behaviors.

Potential Impacts. Failure to diagnose and address these issues can lead to significant misclassifications, security concerns, and overall reduced model *robustness* and *accuracy*. Anomalous behaviors can undermine the trustworthiness and reliability of the ML-enabled system, leading to potential failures in real-world applications.

Target Research. Addressing this challenge involves the development of methods for exploration, *interpretability*, and *explainability* to diagnose failures and misclassifications. Researchers in *fault localization* can contribute by identifying precise locations of faults within the model. *Explainable AI* researchers can focus on creating methods that make the decision-making process of models transparent, aiding in the identification of the root causes of failures. *Software metrics* researchers can develop novel metrics that go beyond *accuracy* to provide actionable insights into the model's *robustness* and reliability. Finally, *automatic program repair* researchers can design tools to automatically fix identified issues within the models, ensuring their proper functioning and enhancing overall system reliability.

- **Challenge C.3 - Context-based Trade-off Identification [S34, S37, S49, S49, S50, S126].** According to our analysis, this challenge affects *"Accuracy"*, *"Resiliency"*, *"Sustainability"*, *"Maintainability"*, and *"Efficiency"*, and in particular *accuracy*, *replaceability*, *security*, *fairness*, and *performance* of ML-enabled systems; however, it is important to note that the challenge may also extend to other NFRs beyond those for which we observed explicit evidence in the reviewed studies:

  Nature of the Challenge. Identifying a trade-off between multiple non-functional requirements, particularly those influenced by specific contextual factors (e.g., regulatory requirements, ethical considerations, and operational constraints), represents a key socio-technical challenge. This step involves exploring and diagnosing which NFRs are most relevant for the deployment environment.

  Possible Underlying Causes. ML-enabled systems must always preserve *accuracy*, but other non-functional requirements such as *robustness*, *security*, or *fairness* may be more or less relevant based on the specific context where the system is deployed. The underlying causes include the socio-technical influences of the deployment environment, such as domain-specific regulations, user expectations, and ethical considerations, which dictate the importance of different NFRs and their trade-offs.

  Potential Impacts. The failure to appropriately identify non-functional requirements based on contextual needs can lead to suboptimal system performance and may even result in critical failures. For instance, a driving system that does not identify and understand specific non-functional requirements such as *robustness* and *security* might be prone to failures or attacks, while a healthcare system that does not emphasize *fairness* might result in biased treatment outcomes. This challenge is particularly relevant for

system analysts, domain experts, and requirements engineers who need to explore and document these trade-offs as an early step in system design.

Target Research. This challenge emphasizes the need for novel approaches to identify non-functional requirements in specific contexts. Researchers in *software analytics* can develop methods to (semi-)automatically mine contextual information, providing practitioners with insights into the non-functional requirements that should be identified and highlighted. Additionally, research in *software project management* can focus on novel methods to assess the significance of non-functional requirements, supporting managerial decisions in the design and deployment of ML-enabled systems. By integrating socio-technical analyses and advanced project management techniques, researchers can contribute to more effective trade-off identification and decision-making processes.

- **Challenge C.4 - Prioritizing and Balancing Non-Functional Requirements [S1, S4, S5, S11, S32, S37, S38, S51, S64, S86, S100, S104].** This challenge affects *"Accuracy"*, *"Resiliency"*, *"Sustainability"*, *"Maintainability"*, and *"Efficiency"*, and in particular non-functional requirements concerning *accuracy*, *robustness*, *replaceability*, *cost*, and *performance* of ML-enabled systems. Similarly to **C.3**, it is likely that this challenge also impacts other NFRs beyond those explicitly discussed in the reviewed studies. The NFRs listed here reflect those most frequently and explicitly linked to this challenge in the primary studies, and they represent areas where balancing priorities has been particularly emphasized. However, given the broad and interconnected nature of this challenge, it is plausible that its relevance extends to additional NFRs that were not directly observed in our analysis:

  Nature of the Challenge. Balancing multiple non-functional requirements represents a key challenge in developing ML-enabled systems. While **C.3** focuses on identifying relevant trade-offs based on context, **C.4** addresses the prioritization and resolution of conflicts between competing NFRs to achieve a balanced system design. This step involves strategic decision-making and technical resolution.

  Possible Underlying Causes. The complexity of balancing non-functional requirements arises from the need to address multiple, often conflicting, objectives simultaneously. Different requirements may have varying levels of importance depending on the specific use case and context, and prioritizing them effectively requires a deep understanding of their interdependencies and impacts. Unlike the exploratory nature of **C.3**, **C.4** involves decision-makers such as software architects and project managers, who must consider trade-offs identified earlier and resolve them to achieve a cohesive system design.

  Potential Impacts. Inadequate prioritization and balancing of non-functional requirements can lead to suboptimal system performance, increased costs, and potential failures. For example, overemphasizing *performance* at the expense of *security* may result in vulnerabilities while focusing too much on *fairness* might reduce *efficiency*. These imbalances can compromise the overall quality and reliability of ML-enabled systems.

  Target Research. The *requirements engineering* research community can address this challenge by proposing guidelines or automated tools to prioritize and balance non-functional requirements. Researchers can develop methods to analyze contextual factors and practitioners' preferences, enabling them to find an optimal balance among correlated and contrasting objectives. This may involve leveraging multi-objective opti-

mization techniques, decision-making frameworks, and tools that assist stakeholders in navigating complex trade-offs and prioritization challenges.

- **Challenge C.5 - Integration and Interaction of ML Models [S61, S65, S70, S95, S118].** This challenge affects *"Accuracy"*, *"Resiliency"*, *"Sustainability"*, *"Maintainability"*, and *"Efficiency"*. Specifically, *accuracy*, *security*, *adaptability*, *cost*, and *performance* of ML-enabled systems:

  Nature of the Challenge. Integrating ML models within complex systems involves synchronizing multiple components, such as traffic light recognition, lane detection, and obstacle perception. Each of these components may utilize several ML models that must function cohesively, hence further complicating the integration.

  Possible Underlying Causes. The dynamic interactions among these ML models create dependencies that are difficult to predict and manage. While individual ML models can undergo rigorous testing, comprehensive testing for the integrated system as a whole is often inadequate. In addition, designing complex architectures requires knowledge of the characteristics of ML models and, perhaps more importantly, of the whole set of non-functional requirements expected to be implemented by the ML-enabled system.

  Potential Impacts. Inadequate integration can lead to unreliability, where the system might fail to perform as expected under certain conditions. This is particularly critical in safety-critical systems, such as autonomous driving, where integration failures can result in dangerous situations and potentially fatal accidents.

  Target Research. Researchers in the fields of *software architecture* and *software testing* can address this challenge by developing methods to ensure seamless integration and interaction of multiple ML models within complex systems. This may involve creating comprehensive testing frameworks that simulate real-world interactions and dependencies among ML models, ensuring that integrated systems are robust and reliable. Additionally, developing synchronization protocols and automated tools for managing dependencies among ML models can help improve the overall reliability of integrated systems. Finally, software architecture techniques should be made model-aware and non-functional-requirements-aware, meaning that they should incorporate pieces of information that may support software architects in the design of complex infrastructures.

- **Challenge C.6 - Improving Practitioners' Knowledge [S69, S105, S129, S130].** This challenge affects the whole set of non-functional requirements of ML-enabled systems:

  Nature of the Challenge. Understanding how to handle non-functional requirements for ML systems compared to traditional software systems can be problematic. ML systems introduce new non-functional requirements and alter the importance and interpretation of existing ones, as pointed out in $\mathbf{RQ}_1$. Practitioners often struggle to adapt existing knowledge and processes to effectively manage these evolving requirements.

  Possible Underlying Causes. Non-functional requirements for ML systems are highly dependent on the application domain and require specific skills for accurate definition and measurement. Additionally, there is a significant lack of awareness and understanding of non-functional requirements among both customers and engineers, leading to insufficient consideration and implementation. Measuring non-functional requirements is also complex, leading to their omission in analyses.

Potential Impacts. These challenges can negatively impact ML-enabled systems, compromising quality and customer satisfaction, and can lead to regulatory non-compliance. For instance, in autonomous vehicles, non-functional requirements related to safety and robustness are critical to preventing accidents. In health diagnostics, accuracy and interpretability are essential for proper diagnosis and treatment, while in financial systems, fairness, transparency, and security are vital to avoid biased decisions and legal issues.

Target Research. Researchers in the fields of *software engineering education*, *requirements engineering*, and *empirical software engineering* can address this challenge by developing training programs and educational materials that enhance practitioners' understanding of NFRs specific to ML systems. Creating guidelines and best practices for the accurate definition, measurement, and management of non-functional requirements in various application domains can also help. Additionally, tools and frameworks that assist in the identification and evaluation of non-functional requirements during the software development lifecycle can support practitioners in ensuring comprehensive consideration of these requirements.

- **Challenge C.7 - Software Analytics for Non-Functional Requirement Assessment [S12, S13, S26, S120, S121, S121].** This challenge affects the whole set of non-functional requirements of ML-enabled systems:

  Nature of the Challenge. Multiple primary studies advocate the need to empower requirements engineering processes with software analytics instruments able to assess the implications that trade-off choices may have on the development of ML-enabled systems.

  Possible Underlying Causes. Current literature highlights a gap in software metrics that can inform practitioners about how their requirements engineering decisions may impact both the complexity of the development process and the overall quality of the system. There is a lack of strategies to recommend quality assurance mechanisms based on specific trade-offs and predictive analytics tools that can enhance practitioners' capabilities in assessing the impact of trade-off analysis.

  Potential Impacts. Without effective software analytics, practitioners may struggle to make informed decisions regarding non-functional requirements, leading to suboptimal trade-offs that could compromise system quality and development efficiency. For example, lacking tools to predict how certain trade-offs will affect the system may result in overlooked quality assurance mechanisms, increasing the risk of defects and reducing system reliability and performance.

  Target Research. Researchers in *requirements engineering* and *software analytics* can address this challenge by developing novel software metrics and analytics tools that help practitioners assess the impact of their decisions regarding non-functional requirements. Collaborative efforts could focus on creating predictive analytics instruments that enhance the ability to foresee the consequences of trade-offs throughout the software lifecycle. By doing so, the research community can improve the support available to requirements engineers, enabling them to make more informed and effective decisions.

- **Challenge C.8 - Improving the Generalizability of Existing Automated Approaches [S19, S36, S50, S53].** This challenge impacts all the non-functional requirements of ML-enabled systems:

Nature of the Challenge. The challenge concerns the generalizability of existing approaches that support the sustainability analysis of ML-enabled systems. Current methods often have limitations that hinder their broader application. Limited generalizability often leads to the development of narrowly applicable tools, which increases effort, reduces the ability to reuse components, and hinders the adoption of certain practices in both the short and long term.

Possible Underlying Causes. Multiple primary studies identified limitations in terms of learning tasks and models support. For instance, recommendation systems like PLUM are restricted to deep learning models, while other approaches can only handle a limited set of deep neural networks, are built using limited datasets [S36], or are not integrated within MLOps pipelines [S19].

Potential Impacts. These limitations restrict the usefulness of current tools in practical scenarios, making it difficult for practitioners to generalize their findings across different types of ML models and learning tasks. Consequently, this can lead to inefficiencies and increased effort in model repair, development, and maintenance, ultimately compromising the sustainability and performance of ML-enabled systems.

Target Research. Researchers in the fields *software analytics*, *machine learning engineering*, and *MLOps* are called to the development of more generalizable, accurate, and integrated tools that support practitioners throughout the software lifecycle. This includes creating instruments that can handle a wide range of neural network architectures, utilize diverse and comprehensive datasets, and seamlessly integrate with MLOps pipelines. Such advancements will enhance the applicability and effectiveness of sustainability analysis tools in real-world settings, thereby improving the overall quality and efficiency of ML-enabled systems.

As a final note, it is worth remarking that we observed that some challenges were discussed by more papers, while others have a lower frequency of mention across the literature. In this respect, we cannot speculate on whether this is because some challenges are inherently more important than others. The number of referenced publications discussing a particular challenge may reflect several factors, such as current research trends, the ease of identifying and addressing certain challenges, or even the availability of data and case studies related to specific issues. In other terms, while a higher number of references might suggest that a challenge is widely recognized or frequently encountered, it does not necessarily indicate its overall importance relative to other challenges. More comprehensive evaluations, considering both quantitative and qualitative aspects, may be necessary to understand the full scope and significance of the challenges associated with non-functional requirements in ML-enabled systems.

☞ **Answer to RQ_2.** We elicited 26 software engineering challenges targeting the individual categories of non-functional requirements identified in our systematic synthesis work and the cross-cutting aspects affecting non-functional requirements engineering. We methodically untangled the challenges faced by ML-enabled systems, providing insights into the specific research fields interested in those issues. According to our results, we call for comprehensive analyses and approaches that may assess the impact and implications of managing individual and multiple non-functional requirements.

5. Discussion and Implications

The results of our study provide a number of additional discussion points and implications, which we discuss further in the following.

Impact of our findings on research and practice. Our results allowed us to synthesize the currently available knowledge on non-functional requirements of ML-enabled systems, hence contributing to a unified framework that consolidates the state of the art in the field [1, 23, 24, 67]. By leveraging the set of primary studies collected through our hybrid systematic literature review, we could aggregate the findings obtained in the broad field of requirements engineering, hence reporting a comprehensive overview of the multifaceted aspects impacting the trustworthiness of ML-enabled systems. We could specifically classify 31 distinct non-functional requirements, categorized into six main classes. More importantly, with $\mathbf{RQ}_1$, we could provide insights into the meaning of the collected non-functional requirements, finding commonalities and peculiarities with respect to non-functional requirements considered in non-ML-enabled systems. For instance, we observed that certain requirements like performance efficiency and traceability have similar interpretations in both ML and non-ML contexts. At the same time, unique non-functional requirements, e.g., retrainability, replaceability, or stability, emerged as aspects that characterize ML-enabled systems and that, therefore, require focused investigations by the research community. Additionally, our classification revealed that a number of non-functional requirements, e.g., scalability and security, present specific nuances within ML-enabled environments, highlighting the need for tailored approaches and methodologies to effectively address these requirements. Through this detailed classification and analysis, our work not only enhances the understanding of non-functional requirements in ML-enabled systems but also provides a reference framework for research and practice. For instance, we envision studies that delve deeper into the unique non-functional requirements of ML-enabled systems in an attempt to characterize their impact and criticality in various application domains. In this respect, the results of $\mathbf{RQ}_1$ inform the research community on the application domains targeted by researchers so far, possibly indicating areas that might be worth investigating. Furthermore, we may also envision studies aiming at refining current requirements elicitation and analysis practices to take the peculiarities of ML-enabled systems into account, e.g., by developing guidelines for eliciting sustainability requirements. Last but not least, our classification may serve as a starting point for researchers to devise novel methodologies for assessing, validating, and monitoring non-functional requirements of ML-enabled systems, e.g., by proposing methods to assess retrainability, replaceability, and stability throughout the lifecycle of ML-enabled systems. From the practitioners' perspective, our classification provides a unified body of knowledge to gather information about the qualities to meet when developing trustworthy ML-enabled systems, raising their awareness on the complex set of non-functional requirements that may impact the development.

› **Take Away Message.** Our classification of non-functional requirements provides a reference framework that comprehensively integrates the current knowledge on non-functional requirements of ML-enabled systems, presenting commonalities with research on non-functional requirements of non-ML-enabled systems and peculiarities that make ML-enabled systems different. We argue that the results to $\mathbf{RQ}_1$ may inspire further research, other than making practitioners aware of the multiple, multi-faceted concerns arising when developing ML-enabled software systems.

When it turns to **RQ**$_2$, our work aggregates a set of 26 software engineering challenges proposed in the literature (e.g., [24, 27]), hence presenting a *super-set* of the challenges documented in earlier research. Through the analysis of the primary studies collected in the hybrid systematic literature review, we could classify the unique challenges specific to the individual classes of non-functional requirements identified in **RQ**$_1$, but also the cross-cutting challenges that affect the whole set of non-functional requirements of ML-enabled systems. From a scientific perspective, our work addresses one of the major challenges identified by Villamizar et al. [67], which was actually related to the lack of systematic knowledge of non-functional requirements of ML-enabled systems. In addition, our work classifies the documented relations between multiple non-functional requirements, which may be seen as an additional contribution that may stimulate further research on the matter. From a tangible perspective, our work has critical implications for multiple research fields. In response to **RQ**$_2$, we could identify the target audience of each software engineering challenge, providing insights that are not only relevant for requirements engineering research but that extend to other sub-communities like software analytics, software testing, and empirical software engineering. Our synthesis effort may pave the way for interactions and joint collaborations toward addressing the many specific challenges of non-functional requirements of ML-enabled systems that encompass analysis, measurement, and monitoring from requirements elicitation till testing and post-deployment operations. Furthermore, the challenges reported by our work may be of interest to professional ML engineers and project managers, who may have an improved understanding of how these challenges may impact the development of real-world systems, other than of the possible obstacles to technological transfer they should preliminary address.

› **Take Away Message.** Our work could identify several software engineering challenges that affect the management of non-functional requirements of ML-enabled systems. We argue that the challenges discussed in **RQ**$_2$ inform future research efforts and possibly stimulate interactions and collaborations among software engineering researchers studying solutions to support practitioners throughout the development lifecycle. In addition, the challenges reported in our work may inform practitioners of the obstacles to the development of trustworthy ML-enabled systems, leading them to more careful considerations of non-functional requirements in real-world contexts.

In addition to the points discussed above, our findings from **RQ**$_1$ have the potential to significantly influence practical applications by helping to refine standards and guidelines for handling non-functional requirements of machine learning systems. For instance, our detailed definitions could enhance emerging standards such as the *ISO/IEC 25059*,[14] which already outlines a taxonomy of non-functional properties. This could lead to more comprehensive guidelines that better support researchers and practitioners in the field. On the one hand, these standards cover various system and software quality attributes, including performance efficiency, reliability, usability, security, maintainability, and portability. Our taxonomy builds on these aspects by introducing ML-specific non-functional requirements such as scalability, retrainability, interpretability, and explainability, which are critical because of the ML-enabled systems' dynamic and data-intensive nature. On the other hand, Habibullah et al. [23] focused mainly on ethics and biases within non-functional requirements for ML systems. Our study expands this perspective to include other attributes such as energy consumption, sustainability,

[14]The *ISO/IEC 25059* standard: `https://iso25000.com/index.php/en/iso-25000-standards/iso-25059`

accountability, and robustness. This comprehensive approach ensures coverage of all critical quality aspects of ML systems, providing a more complete framework for assessing and managing non-functional requirements in these systems. In other terms, our consolidated classification was therefore needed to synthesize the currently emerging knowledge on the specific characteristics and requirements of ML-enabled systems - aspects that are not fully covered by existing standards and classifications. Furthermore, our results complement the insights from Amershi et al.[5], which identified best practices for the actionable development of machine learning systems, including aspects like model debugging and interpretability. Our work provides concrete factors that directly relate to these practices, such as the *retrainability* attribute within the *'Maintainability'* category, offering a foundation for developing measurable and monitorable systems through the various stages of machine learning system development. In addition, the findings of $\mathbf{RQ}_2$ provide insights into the research gaps, making the best practices identified by Amershi et al. [5] difficult to actually implement. Also in this case, our systematic synthesis may provide researchers with an improved understanding of the next steps to pursue better support for the development and evolution of ML-enabled systems. Ali et al. [4] emphasized that quality standards should follow the 'ility' procedure based on traditional quality models and require comprehensive quality models for artificial intelligence software components. In this respect, our work could greatly contribute to the creation of new quality standards that would be more robust and tailored to the complexities of AI software, improving its reliability and effectiveness.

> **Take Away Message.** Our findings complement the current state of the practice in different manners. First, the outcome of $\mathbf{RQ}_1$ may be useful to extend and/or complement emerging standards describing relevant non-functional requirements of machine learning-enabled systems. Second, the results of $\mathbf{RQ}_1$ and $\mathbf{RQ}_2$ can be combined with the pieces of information that emerged from the state of the practice, highlighting the current research gaps that should be filled and the potential opportunities of technological transfer.

On the Inter-Relation among Non-Functional Requirements. Based on the cross-cutting challenges discussed in the context of $\mathbf{RQ}_2$, we could identify the presence of some inter-relations among non-functional requirements, i.e., how the satisfaction or optimization of one non-functional requirement can influence or be influenced by another. These are not only concerned with the relation that accuracy has with other requirements but also with the innate interconnections between multiple non-functional aspects playing a role in developing ML-enabled systems [S54, S64, S99]. Our results demonstrated that groups within the same cluster are closely related, making it easier to improve others within the same cluster when one is enhanced [S72, S110, S122]. Conversely, improving one cluster can lead to the deterioration of non-functional requirements in different clusters [S11, S86, S95]. In this respect, researchers in requirements engineering and empirical software engineering might cooperate toward developing novel taxonomies that may map the relations among the non-functional requirements of ML-enabled systems and how they impact each other. Our results also called for further research on identifying, managing, and assessing the trade-offs among multiple non-functional requirements.

> **Take Away Message.** Our findings suggest novel interconnections and research on the multi-objective optimization of non-functional requirements of ML-enabled systems. We call for a new research field focusing on empirically investigating the relations between non-functional requirements and how those relations may inform the development of automated approaches to optimize ML-enabled systems.

A Managerial Viewpoint. As a follow-up discussion, our research questions raise multiple trade-offs to consider when developing ML-enabled systems. Our work has implications on the managerial side: the constant and continuous need to measure non-functional requirements and search for trade-offs represent high-level challenges for project managers, who are required to monitor and handle multiple non-functional attributes throughout the evolution of ML-enabled systems. In the first place, the results produced by $\mathbf{RQ}_1$ may allow the reader to understand what the set of non-functional attributes actually takes into account. Secondly, the challenges identified in the context of $\mathbf{RQ}_2$ should not only be considered from a technical perspective but also from a socio-technical and managerial one. Our work calls for further research on the managerial strategies that would indicate the most appropriate management policies to apply when handling non-functional requirements.

> **Take Away Message.** We call for research on managerial and socio-technical strategies to handle non-functional requirements throughout the evolution of ML-enabled systems. At the same time, our findings point out the need for optimization approaches specifically tailored to software evolution, hence being contextual, dependent, and adaptable.

6. Threats to Validity

Our systematic literature review encountered potential limitations, which we mitigated through careful design. These limitations mainly pertain to the methodology employed to collect and analyze the primary studies object of the study, as elaborated in the following sections.

Literature selection. The article selection steps posed significant challenges due to the inherent variability in the terminology used by primary studies to refer to non-functional requirements of ML-enabled systems. In fact, when analyzing the results coming from the initial search string we realized that a significant number of relevant articles may have referred to specific non-functional requirements, e.g., fairness, rather than to the general concept of non-functional requirements. This aspect may have negatively impacted the comprehensiveness of our systematic literature review. We mitigated such a potential limitation by enlarging the scope of our work, designing a *hybrid* literature review that combined standard guidelines [32] with additional steps such as screening top-tier venues [74] and snowballing [72]. In our seed search methodology, we focused on scanning articles published in A* and A conferences and Q1 journals. The decision to limit our search to these high-impact venues stems from their recognition in the research community for rigorous review processes and high relevance. Nonetheless, we acknowledge that this approach may have excluded relevant studies from other reputable conferences and journals in the area of requirements engineering. While this remains a limitation of the study, we mitigated the risk of excluding relevant articles by including (1) the major requirements engineering conference (RE) and (2) top-tier software engineering conferences and journals, e.g., ICSE, TSE, TOSEM, that regularly publish requirements engineering research.

Still discussing the limitations of the seed search methodology, we considered expanding the scope of the analysis to include a subset of AI venues. This decision was based on the recognition that AI researchers might have addressed non-functional requirements within the context of ML-enabled systems, providing valuable insights relevant to our study. However, achieving comprehensive coverage of the most pertinent AI venues would have been nearly impossible, as it would have required surveying research articles published between 2012 and 2023 across approximately 70 Q1 journals and over 40 A* and A conferences. To make the seed search process more sustainable, we analyzed a subset of top-tier venues in the field, focusing on those with a higher likelihood of including relevant engineering work. Among the papers published in the selected venues between 2012 and 2023, we found only six potentially relevant articles, which were further reduced to two after the quality assessment stage. We acknowledge that an exhaustive analysis of AI venues might have identified additional primary studies, and this remains a limitation of our work that the reader should be aware of. However, our analysis revealed that the number of papers in AI venues describing non-functional requirements is quite limited (only 0.04% of the papers scanned were finally included), possibly not justifying the significant effort that would have been required. Alternative research methods, such as qualitative surveys or interviews, might be more effective in complementing our findings.

In the second place, it is worth remarking that we performed the search on multiple databases such as *ACM Digital Library*, *Scopus*, and *IEEEXplore*. This was done to ensure wider coverage of the primary studies published in the literature.

The reliance on a hybrid systematic literature review, other than the multiple actions conducted to extend the search process, makes us confident of the completeness of the literature selection. However, for the sake of verifiability and replicability, our online appendix [45] contains all the data and material used to produce each intermediate search of our study. The interested reader might use the material to either assess the soundness/completeness of the process and further build on top of our results [74].

Literature analysis and synthesis. We applied exclusion and inclusion criteria and performed quality assessments manually, which might have introduced subjectivity and human error. We mitigated this risk through two main actions. First, we crafted the criteria for determining the suitability of the collected articles to be as objective as possible, clearly defining the specific pieces of information that articles must contain to be considered for analysis. More specifically, we filtered papers based on their relevance to the topic, ensuring that only papers explicitly addressing non-functional requirements in ML-enabled systems were considered. These papers need to focus on specific types of non-functional requirements, describe associated challenges, and specify relevant domains. These stringent criteria enabled us to significantly reduce the number of candidate papers to a manageable amount. We excluded articles that did not contextualize their findings within the framework of non-functional requirements in ML-enabled systems. For example, papers that focused on the technical aspects of a proposed methodology without explaining how the methodology addressed specific non-functional requirements of ML-enabled systems were not considered for inclusion. On the one hand, this minimized the risk of subjective interpretation. On the other hand, we also defined quality assessment criteria to evaluate the quality, depth, and contribution of each paper that successfully passed the inclusion criteria. This additional step ensured that only high-quality studies were included in our analysis, further reducing the risk of bias and enhancing the reliability of our findings. Of course, we are aware that some relevant primary studies might have been missed. This represents an inherent limitation of our review process. However, we believe that the rigorous application of our criteria has

enabled us to provide a comprehensive and reliable synthesis of the current state of research on non-functional requirements in ML-enabled systems.

Perhaps more importantly, we also conducted a two-step validation of the selection process. We specifically retained a sample of 100 articles to be used as a validation set. Both authors of the paper independently reviewed these papers, applying the exclusion/inclusion criteria and quality assessment criteria. The resulting set of papers included by the two authors was then compared to identify differences in terms of the application of the criteria. Upon discussion, they refined the evaluation process by clarifying the criteria and resolving any ambiguities, i.e., this initial comparison and subsequent discussion helped align their understanding and application of the criteria. Subsequently, they conducted a second round of reviews on an additional set of 50 papers to verify the consistency of their refined criteria. This additional round resulted in a full agreement between the authors, thus confirming the effectiveness of the refined criteria. This established a solid foundation for the first author to proceed confidently with the assessment of the entire set of papers and further reduced the risk of the subjectiveness of the analysis process.

In terms of synthesis of the primary studies, we basically employed a qualitative content analysis process through which we could (1) extract pieces of information useful to address our research questions by filling the data extraction form for each article; (2) synthesize the data coming from the primary studies; and (3) report the synthesized information.

In the context of $\mathbf{RQ}_1$, we focused on the extraction and classification of non-functional requirements. We began by analyzing the primary studies to identify and extract explicitly mentioned non-functional requirements. To mitigate potential threats to validity, we ensured a thorough reading and annotation of relevant sections. This process minimized the risk of overlooking critical information and enhanced the reliability of our findings. Each requirement was assigned a unique name and detailed description, with inconsistencies resolved to ensure clarity. Mapping the non-functional requirements to high-level reference classes further structured our findings, providing a coherent framework for understanding the requirements. As for $\mathbf{RQ}_2$, we focused on identifying and synthesizing the challenges associated with non-functional requirements in ML-enabled systems. In this respect, two clarifications would be worth remarking. In the first place, we intentionally focused on challenges that directly influence the achievement, management, or optimization of specific NFRs in ML-enabled systems. Rather than attempting to include all challenges faced by ML systems, we concentrated solely on those explicitly connected to NFRs. These challenges were identified exclusively from the primary studies included in our systematic literature review, ensuring that every challenge discussed is traceable to the evidence presented in the reviewed papers. As a result, challenges that were not explicitly discussed in these studies, even if potentially relevant to NFRs, were excluded to preserve the methodological rigor and defined scope of our analysis. Secondly, we included only those challenges that were explicitly linked to NFRs in the reviewed studies. Although the primary studies sometimes addressed a broader range of issues, our focus was on extracting and synthesizing challenges that directly pertain to NFRs. For instance, in studies exploring algorithmic discrimination, we included challenges such as mitigating disparate impacts or ensuring equitable model performance, as these are intrinsically tied to the NFRs of fairness and accuracy. Conversely, challenges unrelated to NFRs, such as purely technical optimizations without a clear connection to NFRs, were excluded. Similarly, we excluded general ML system challenges, such as hyperparameter tuning or convergence improvements, unless the primary studies explicitly linked these to their impact on specific NFRs. This deliberate focus ensured that the challenges discussed remained within the scope of NFR-related considerations in ML-enabled systems. More in general, we extracted information about the nature of the challenges, their underlying causes, and their potential impacts. To mit-

igate threats to validity, we grouped similar challenges together and identified common themes, ensuring a comprehensive synthesis of the data. The synthesized challenges were reviewed and validated through discussions between the authors, which helped ensure consistency and accuracy in categorization.

In other terms, we established data analysis procedures in a way that reduced the risk of inconsistencies, minimized subjective bias, and altogether enhanced the reliability of our findings. To make our process as transparent as possible, we documented the processes in detail and released all the material used in systematic literature as part of our online appendix [45].

The entire analysis and synthesis process was grounded in the authors' expertise in non-functional requirements and ML-enabled systems. The two authors have a research experience of two and eleven years, respectively. Both conduct or have already conducted quantitative and qualitative studies in the past, other than systematic literature/mapping studies on themes connected to software engineering for artificial intelligence, artificial intelligence for software engineering, software quality, and software maintenance and evolution. The authors have been actively engaged in the analysis of non-functional requirements of ML-enabled systems. The first author's Ph.D. dissertation focuses on optimizing non-functional requirements of ML-enabled systems. The second author has published articles on this topic and currently coordinates the research activities of six Ph.D. students who are working on the analysis and optimization of non-functional requirements for ML-enabled systems. Furthermore, he serves as the Local Coordinator and Principal Investigator of two national projects centered around the themes explored in this article, which have partially supported this research. Furthermore, they are both involved in the academic courses of *Software Engineering*, *Fundamentals of Artificial Intelligence*, and *Software Engineering for Artificial Intelligence* at the University of Salerno (Italy) with the second author serving as the lecturer for these courses, while the first author acts as a teaching assistant. The expertise accumulated through these research and educational activities reduced the risk of subjectivity and/or wrong application of data analysis and synthesis protocols.

7. Conclusion and Future Work

This paper presents a hybrid systematic literature review on non-functional requirements of machine learning-enabled systems. We focused on two key areas: classifying non-functional requirements and identifying associated challenges. Our review not only summarizes existing knowledge but also highlights new horizons and challenges for the research community to explore. We hope this work inspires further contributions from researchers and Ph.D. students.

To sum up, our study provides three major contributions:

1. A systematic review categorizing non-functional attributes of machine learning systems and outlining current challenges, aiding researchers in defining future research directions.

2. Implications and key messages for researchers to address future research avenues on managing and optimizing non-functional requirements.

3. An online appendix with data and scripts used in the study, which facilitates reproducibility, replicability, and further extension of our work.

The findings of the study, as well as the challenges identified in our work, guide our future research, where we aim to delve deeper into how the research community addresses non-functional

requirements and explore novel approaches to optimize them while maintaining accuracy. Future research can build on top of our work to identify non-functional metrics to gain a deeper understanding of these aspects, thereby improving the manageability of ML-enabled systems for researchers and practitioners. Additionally, we plan to conduct software analytics studies, surveys, and interviews to understand the significance of the challenges associated with non-functional requirements and their impact on practitioners' daily activities. Future research may also focus on understanding the different non-functional requirements of Large Language Models and Generative AI, which present unique challenges such as hallucination, sustainability, and ethical concerns [44]. In this sense, our work might be expanded by developing a new taxonomy based on the specific non-functional requirements of these emerging new technologies.

Credits

Vincenzo De Martino: Formal analysis, Investigation, Data Curation, Validation, Writing - Original Draft, Visualization. **Fabio Palomba**: Supervision, Validation, Writing - Review & Editing.

Conflict of interest

The authors declare that they have no conflict of interest.

Data Availability

The data collected in the context of this systematic literature review, along with the scripts used to analyze and generate data, charts, and plots discussed when addressing our research goals, are publicly available at: [45]

Acknowledgement

This work has been partially supported by the European Union - NextGenerationEU through the Italian Ministry of University and Research, Projects PRIN 2022 "QualAI: Continuous Quality Improvement of AI-based Systems" (grant n. 2022B3BP5S , CUP: H53D23003510006) and PRIN 2022 PNRR "FRINGE: context-aware FaiRness engineerING in complex software systEms" (grant n. P2022553SL, CUP: D53D23017340001). The authors would like to thank the associate editor and anonymous reviewers for their recommendations and suggestions, which were instrumental in improving the soundness and overall quality of our manuscript.

References

[1] Khlood Ahmad, Mohamed Abdelrazek, Chetan Arora, Muneera Bano, and John Grundy. 2023. Requirements engineering for artificial intelligence systems: A systematic mapping study. *Information and Software Technology* 158 (2023), 107176.

[2] Khlood Ahmad, Mohamed Abdelrazek, Chetan Arora, Muneera Bano, and John Grundy. 2023. Requirements practices and gaps when engineering human-centered Artificial Intelligence systems. *Applied Soft Computing* 143 (2023), 110421.

[3] Asad Ali and Carmine Gravino. 2019. A systematic literature review of software effort prediction using machine learning methods. *Journal of Software: Evolution and Process* 31, 10 (2019), e2211. `https://doi.org/10.1002/smr.2211` arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/smr.2211 e2211 JSME-18-0247.R2.

[4] Mohamed Abdullahi Ali, Ng Keng Yap, Abdul Azim Abd Ghani, Hazura Zulzalil, Novia Indriaty Admodisastro, and Amin Arab Najafabadi. 2022. A systematic mapping of quality models for AI systems, software and components. *Applied Sciences* 12, 17 (2022), 8700.

[5] Saleema Amershi, Andrew Begel, Christian Bird, Robert DeLine, Harald Gall, Ece Kamar, Nachiappan Nagappan, Besmira Nushi, and Thomas Zimmermann. 2019. Software engineering for machine learning: A case study. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. IEEE, 291–300.

[6] Muhammad Ilyas Azeem, Fabio Palomba, Lin Shi, and Qing Wang. 2019. Machine learning techniques for code smell detection: A systematic literature review and meta-analysis. *Information and Software Technology* 108 (2019), 115–138.

[7] Osbert Bastani, Yani Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya Nori, and Antonio Criminisi. 2016. Measuring Neural Net Robustness with Constraints. In *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Inc. `https://proceedings.neurips.cc/paper/2016/file/980ecd059122ce2e50136bda65c25e07-Paper.pdf`

[8] Christoph Becker, Ruzanna Chitchyan, Leticia Duboc, Steve Easterbrook, Birgit Penzenstadler, Norbert Seyff, and Colin C Venters. 2015. Sustainability design and software: The karlskrona manifesto. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Vol. 2. IEEE, 467–476.

[9] Manal Binkhonain and Liping Zhao. 2019. A review of machine learning algorithms for identification and classification of non-functional requirements. *Expert Systems with Applications: X* 1 (2019), 100001.

[10] Markus Borg, Cristofer Englund, Krzysztof Wnuk, Boris Duran, Christoffer Levandowski, Shenjian Gao, Yanwen Tan, Henrik Kaijser, Henrik Lönn, and Jonas Törnqvist. 2018. Safely entering the deep: A review of verification and validation for machine learning and a challenge elicitation in the automotive industry. *arXiv preprint arXiv:1812.05389* (2018).

[11] Houssem Ben Braiek and Foutse Khomh. 2020. On testing machine learning programs. *Journal of Systems and Software* 164 (2020), 110542.

[12] Bernd Bruegge and Allen H Dutoit. 2009. Object–oriented software engineering. using uml, patterns, and java. *Learning* 5, 6 (2009), 7.

[13] Yuriy Brun and Alexandra Meliou. 2018. Software fairness. In *Proceedings of the 2018 26th ACM joint meeting on european software engineering conference and symposium on the foundations of software engineering*. 754–759.

[14] Arina Cheverda, Ahror Jabborov, Artem Kruglov, and Giancarlo Succi. 2022. State-of-the-Art Review of Taxonomies for Quality Assessment of Intelligent Software Systems. In *2022 3rd International Informatics and Software Engineering Conference (IISEC)*. IEEE, 1–6.

[15] Reza Damirchi and Amineh Amini. 2023. Non-Functional Requirement Extracting Methods for AI-based Systems: A Survey. In *2023 13th International Conference on Computer and Knowledge Engineering (ICCKE)*. IEEE, 535–539.

[16] Elder Vicente de Paulo Sobrinho, Andrea De Lucia, and Marcelo de Almeida Maia. 2018. A systematic literature review on bad smells–5 w's: which, when, what, who, where. *IEEE Transactions on Software Engineering* 47, 1 (2018), 17–66.

[17] Jesse Dillard, Veronica Dujon, and Mary C King. 2008. *Understanding the social dimension of sustainability*. Routledge.

[18] Simone Disabato and Manuel Roveri. 2022. Tiny machine learning for concept drift. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[19] Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou. 2017. Fairness testing: testing software for discrimination. In *Proceedings of the 2017 11th Joint meeting on foundations of software engineering*. 498–510.

[20] Bahar Gezici and Ayça Kolukısa Tarhan. 2022. Systematic literature review on software quality for AI-based software. *Empirical Software Engineering* 27, 3 (2022), 66.

[21] Görkem Giray. 2021. A software engineering perspective on engineering machine learning systems: State of the art and challenges. *Journal of Systems and Software* 180 (2021), 111031.

[22] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. BadNets: Evaluating Backdooring Attacks on Deep Neural Networks. *IEEE Access* 7 (2019), 47230–47244. `https://doi.org/10.1109/ACCESS.2019.2909068`

[23] Khan Mohammad Habibullah, Gregory Gay, and Jennifer Horkoff. 2022. Non-functional requirements for machine learning: An exploration of system scope and interest. In *2022 IEEE/ACM 1st International Workshop on Software Engineering for Responsible Artificial Intelligence (SE4RAI)*. IEEE, 29–36.

[24] Khan Mohammad Habibullah, Gregory Gay, and Jennifer Horkoff. 2023. Non-functional requirements for machine learning: Understanding current use and challenges among practitioners. *Requirements Engineering* 28, 2 (2023), 283–316.

[25] Gaétan Hains, Arvid Jakobsson, and Youry Khmelevsky. 2018. Towards formal methods and software engineering for deep learning: security, safety and productivity for dl systems development. In *2018 Annual IEEE international*

systems conference (syscon). IEEE, 1–5.

[26] Lorin Hochstein. 2023. Why Don't We See Even More Failures? *IEEE Software* 40, 4 (2023), 114–116.

[27] Jennifer Horkoff. 2019. Non-functional requirements for machine learning: Challenges and new directions. In *2019 IEEE 27th International Requirements Engineering Conference (RE)*. IEEE, 386–391.

[28] Ling Huang, Anthony D. Joseph, Blaine Nelson, Benjamin I.P. Rubinstein, and J. D. Tygar. 2011. Adversarial Machine Learning. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence* (Chicago, Illinois, USA) *(AISec '11)*. Association for Computing Machinery, New York, NY, USA, 43–58. https://doi.org/10.1145/2046684.2046692

[29] Meenu Mary John, Helena Holmström Olsson, and Jan Bosch. 2021. Architecting AI deployment: a systematic review of state-of-the-art and state-of-practice literature. In *Software Business: 11th International Conference, ICSOB 2020, Karlskrona, Sweden, November 16–18, 2020, Proceedings 11*. Springer, 14–29.

[30] Aaiza Khan, Isma Farah Siddiqui, Mehwish Shaikh, Shabana Anwar, and Murk Shaikh. 2022. Handling non-fuctional requirements in IoT-based machine learning systems. In *2022 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON)*. IEEE, 477–479.

[31] Minsu Kim, Walid Saad, Mohammad Mozaffari, and Merouane Debbah. 2022. On the tradeoff between energy, precision, and accuracy in federated quantized neural networks. In *ICC 2022-IEEE International Conference on Communications*. IEEE, 2194–2199.

[32] Barbara Kitchenham, O Pearl Brereton, David Budgen, Mark Turner, John Bailey, and Stephen Linkman. 2009. Systematic literature reviews in software engineering–a systematic literature review. *Information and software technology* 51, 1 (2009), 7–15.

[33] Barbara Kitchenham and Stuart Charters. 2007. Guidelines for performing Systematic Literature Reviews in Software Engineering. 2 (01 2007).

[34] Maximilian A Köhl, Kevin Baum, Markus Langer, Daniel Oster, Timo Speith, and Dimitri Bohlender. 2019. Explainability as a non-functional requirement. In *2019 IEEE 27th International Requirements Engineering Conference (RE)*. IEEE, 363–368.

[35] Fumihiro Kumeno. 2019. Sofware engneering challenges for machine learning applications: A literature review. *Intelligent Decision Technologies* 13, 4 (2019), 463–476.

[36] Patricia Lago, Sedef Akinli Koçak, Ivica Crnkovic, and Birgit Penzenstadler. 2015. Framing sustainability as a property of software quality. *Commun. ACM* 58, 10 (2015), 70–78.

[37] Guofu Li, Pengjia Zhu, Jin Li, Zhemin Yang, Ning Cao, and Zhiyi Chen. 2018. Security matters: A survey on adversarial machine learning. *arXiv preprint arXiv:1810.07339* (2018).

[38] Ji Lin, Wei-Ming Chen, Yujun Lin, Chuang Gan, Song Han, et al. 2020. Mcunet: Tiny deep learning on iot devices. *Advances in Neural Information Processing Systems* 33 (2020), 11711–11722.

[39] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. 2021. When machine learning meets privacy: A survey and outlook. *ACM Computing Surveys (CSUR)* 54, 2 (2021), 1–36.

[40] Sin Kit Lo, Qinghua Lu, Chen Wang, Hye-Young Paik, and Liming Zhu. 2021. A systematic literature review on federated machine learning: From a software engineering perspective. *ACM Computing Surveys (CSUR)* 54, 5 (2021), 1–39.

[41] Giuliano Lorenzoni, Paulo Alencar, Nathalia Nascimento, and Donald Cowan. 2021. Machine learning model development from a software engineering perspective: A systematic literature review. *arXiv preprint arXiv:2102.07574* (2021).

[42] Lucy Ellen Lwakatare, Aiswarya Raj, Ivica Crnkovic, Jan Bosch, and Helena Holmström Olsson. 2020. Large-scale machine learning systems in real-world industrial settings: A review of challenges and solutions. *Information and software technology* 127 (2020), 106368.

[43] Silverio Martínez-Fernández, Justus Bogner, Xavier Franch, Marc Oriol, Julien Siebert, Adam Trendowicz, Anna Maria Vollmer, and Stefan Wagner. 2022. Software engineering for AI-based systems: a survey. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 31, 2 (2022), 1–59.

[44] Vincenzo De Martino, Joel Castaño, Fabio Palomba, Xavier Franch, and Silverio Martínez-Fernández. 2024. A Framework for Using LLMs for Repository Mining Studies in Empirical Software Engineering. arXiv:2411.09974 [cs.SE] https://arxiv.org/abs/2411.09974

[45] Vincenzo De Martino and Fabio Palomba. 2025. Classification and Challenges of Non-Functional Requirements in ML-Enabled Systems: A Systematic Literature Review. (1 2025). https://doi.org/10.6084/m9.figshare.22815770

[46] Satoshi Masuda, Kohichi Ono, Toshiaki Yasue, and Nobuhiro Hosokawa. 2018. A survey of software quality for machine learning applications. In *2018 IEEE International conference on software testing, verification and validation workshops (ICSTW)*. IEEE, 279–284.

[47] Sean McGuire, Erin Schultz, Bimpe Ayoola, and Paul Ralph. 2023. Sustainability is stratified: Toward a better theory of sustainable software engineering. In *2023 IEEE/ACM 45th International Conference on Software Engineering*

(ICSE). IEEE, 1996–2008.

[48] Claire Cain Miller. 2015. Can an algorithm hire better than a human. *The New York Times* 25 (2015).

[49] Erica Mourão, João Felipe Pimentel, Leonardo Murta, Marcos Kalinowski, Emilia Mendes, and Claes Wohlin. 2020. On the performance of hybrid search strategies for systematic literature reviews in software engineering. *Information and software technology* 123 (2020), 106294.

[50] Elizamary Nascimento, Anh Nguyen-Duc, Ingrid Sundbø, and Tayana Conte. 2020. Software engineering for artificial intelligence and machine learning software: A systematic literature review. *arXiv preprint arXiv:2011.03751* (2020).

[51] MIT News. 2023. Shrinking deep learning's carbon footprint. `https://news.mit.edu/2020/shrinking-deep-learning-carbon-footprint-0807`. Accessed: 2023-11-26.

[52] Phuong T Nguyen, Claudio Di Sipio, Juri Di Rocco, Massimiliano Di Penta, and Davide Di Ruscio. 2021. Adversarial attacks to api recommender systems: Time to wake up and smell the coffee?. In *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 253–265.

[53] Parmy Olson. 2011. The algorithm that beats your bank manager. *CNN Money March* 15 (2011).

[54] Abbas Ourmazd. 2020. Science in the age of machine learning. *Nature Reviews Physics* 2, 7 (2020), 342–343.

[55] Daniele Ravì, Charence Wong, Fani Deligianni, Melissa Berthelot, Javier Andreu-Perez, Benny Lo, and Guang-Zhong Yang. 2016. Deep learning for health informatics. *IEEE journal of biomedical and health informatics* 21, 1 (2016), 4–21.

[56] Jörg Rech and Klaus-Dieter Althoff. 2004. Artificial intelligence and software engineering: Status and future trends. *KI* 18, 3 (2004), 5–11.

[57] Harvard Business Review. 2023. AI Is Not Just Getting Better, it's Becoming More Pervasive. `https://hbr.org/sponsored/2019/02/ai-is-not-just-getting-better-its-becoming-more-pervasive`. Accessed: 2023-03-05.

[58] Vincenzo Riccio, Gunel Jahangirova, Andrea Stocco, Nargiz Humbatova, Michael Weiss, and Paolo Tonella. 2020. Testing machine learning based systems: a systematic mapping. *Empirical Software Engineering* 25 (2020), 5193–5254.

[59] Krishna Ronanki, Beatriz Cabrero-Daniel, Jennifer Horkoff, and Christian Berger. 2023. RE-centric Recommendations for the Development of Trustworthy (er) Autonomous Systems. In *Proceedings of the First International Symposium on Trustworthy Autonomous Systems*. 1–8.

[60] Alex Serban, Koen van der Blom, Holger Hoos, and Joost Visser. 2020. Adoption and effects of software engineering best practices in machine learning. In *Proceedings of the 14th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. 1–12.

[61] Alex Serban and Joost Visser. 2021. An empirical study of software architecture for machine learning. *arXiv preprint arXiv:2105.12422* 39 (2021).

[62] Shriram Shanbhag, Sridhar Chimalakonda, Vibhu Saujanya Sharma, and Vikrant Kaulgud. 2022. Towards a catalog of energy patterns in deep learning development. In *Proceedings of the 26th International Conference on Evaluation and Assessment in Software Engineering*. 150–159.

[63] Yap Yan Siang, Mohd Ridzuan Ahamd, and Mastura Shafinaz Zainal Abidin. 2021. Anomaly detection based on tiny machine learning: A review. *Open International Journal of Informatics* 9, Special Issue 2 (2021), 67–78.

[64] Miltiadis Siavvas, Dimitrios Tsoukalas, Marija Jankovic, Dionysios Kehagias, and Dimitrios Tzovaras. 2022. Technical debt as an indicator of software security risk: a machine learning approach for software development enterprises. *Enterprise Information Systems* 16, 5 (2022), 1824017.

[65] Nuttanai Suwonchoochit and Twittie Senivongse. 2021. Classification of Database Technology Problems on Stack Overflow. In *2021 IEEE/ACIS 19th International Conference on Software Engineering Research, Management and Applications (SERA)*. IEEE, 21–26.

[66] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. `https://doi.org/10.48550/ARXIV.1312.6199`

[67] Hugo Villamizar, Tatiana Escovedo, and Marcos Kalinowski. 2021. Requirements engineering for machine learning: A systematic mapping study. In *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 29–36.

[68] David Wagner and Paolo Soto. 2002. Mimicry Attacks on Host-Based Intrusion Detection Systems. In *Proceedings of the 9th ACM Conference on Computer and Communications Security* (Washington, DC, USA) *(CCS '02)*. Association for Computing Machinery, New York, NY, USA, 255–264. `https://doi.org/10.1145/586110.586145`

[69] Simin Wang, Liguo Huang, Jidong Ge, Tengfei Zhang, Haitao Feng, Ming Li, He Zhang, and Vincent Ng. 2020. Synergy between machine/deep learning and software engineering: How far are we? *arXiv preprint arXiv:2008.05515* (2020).

[70] Xianmin Wang, Jing Li, Xiaohui Kuang, Yu-an Tan, and Jin Li. 2019. The security of machine learning in an adversarial setting: A survey. *J. Parallel and Distrib. Comput.* 130 (2019), 12–23.

[71] Hironori Washizaki, Hiromu Uchida, Foutse Khomh, and Yann-Gaël Guéhéneuc. 2019. Studying software engi-

neering patterns for designing machine learning systems. In *2019 10th International Workshop on Empirical Software Engineering in Practice (IWESEP)*. IEEE, 49–495.

[72] Claes Wohlin. 2014. Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering* (London, England, United Kingdom) *(EASE '14)*. Association for Computing Machinery, New York, NY, USA, Article 38, 10 pages. https://doi.org/10.1145/2601248.2601268

[73] Claes Wohlin. 2016. Second-generation systematic literature studies using snowballing. In *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*. 1–6.

[74] Claes Wohlin, Emilia Mendes, Katia Romero Felizardo, and Marcos Kalinowski. 2020. Guidelines for the search strategy to update systematic literature reviews in software engineering. *Information and software technology* 127 (2020), 106366.

[75] Mengwei Xu, Jiawei Liu, Yuanqiang Liu, Felix Xiaozhu Lin, Yunxin Liu, and Xuanzhe Liu. 2019. A First Look at Deep Learning Apps on Smartphones. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 2125–2136. https://doi.org/10.1145/3308558.3313591

[76] Jie M Zhang and Mark Harman. 2021. " Ignorance and Prejudice" in Software Fairness. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 1436–1447.

[77] Jie M Zhang, Mark Harman, Lei Ma, and Yang Liu. 2020. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering* 48, 1 (2020), 1–36.

[78] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)* 52, 1 (2019), 1–38.

[79] Jianlong Zhou and Fang Chen. 2018. *Human and Machine Learning*. Springer.

SLR – Sistematic Literature Review

[S1] Huiqiang Chen, Tianqing Zhu, Tao Zhang, Wanlei Zhou, and Philip S Yu. Privacy and fairness in federated learning: on the perspective of tradeoff. *ACM Computing Surveys*, 56(2):1–37, 2023.

[S2] Sumon Biswas and Hridesh Rajan. Do the machine learning models on a crowd sourced platform exhibit bias? an empirical study on model fairness. In *Proceedings of the 28th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering*, pages 642–653, 2020.

[S3] Amin Aminifar, Matin Shokri, Fazle Rabbi, Violet Ka I. Pun, and Yngve Lamo. Extremely randomized trees with privacy preservation for distributed structured health data. *IEEE Access*, 10:6010–6027, 2022.

[S4] Wiebke Hutiri, Aaron Yi Ding, Fahim Kawsar, and Akhil Mathur. Tiny, always-on, and fragile: Bias propagation through design choices in on-device machine learning workflows. *ACM Transactions on Software Engineering and Methodology*, 32(6):1–37, 2023.

[S5] Raluca Maria Hampau, Maurits Kaptein, Robin van Emden, Thomas Rost, and Ivano Malavolta. An empirical study on the performance and energy consumption of ai containerization strategies for computer-vision tasks on the edge. In *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering 2022*, EASE '22, page 50–59, New York, NY, USA, 2022. Association for Computing Machinery.

[S6] Hao Zhang and W.K. Chan. Apricot: A weight-adaptation approach to fixing deep learning models. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 376–387, 2019.

[S7] Hao Zhang and W.K. Chan. Plum: Exploration and prioritization of model repair strategies for fixing deep learning models. In *2021 8th International Conference on Dependable Systems and Their Applications (DSA)*, pages 140–151, 2021.

[S8] Lizhi Liao, Heng Li, Weiyi Shang, and Lei Ma. An empirical study of the impact of hyperparameter tuning and model optimization on the performance properties of deep neural networks. *ACM Trans. Softw. Eng. Methodol.*, 31(3), apr 2022.

[S9] Deqing Zou, Yawei Zhu, Shouhuai Xu, Zhen Li, Hai Jin, and Hengkai Ye. Interpreting deep learning-based vulnerability detector predictions based on heuristic searching. *ACM Trans. Softw. Eng. Methodol.*, 30(2), mar 2021.

[S10] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, ASE '18, page 132–142, New York, NY, USA, 2018. Association for Computing Machinery.

[S11] Jie Zhu, Leye Wang, and Xiao Han. Safety and performance, why not both? bi-objective optimized model compression toward ai software deployment. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, ASE '22, New York, NY, USA, 2023. Association for Computing Machinery.

[S12] Simin Chen, Mirazul Haque, Cong Liu, and Wei Yang. Deepperform: An efficient approach for performance testing of resource-constrained neural networks. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, ASE '22, New York, NY, USA, 2023. Association for Computing Machinery.

[S13] Shen Yan, Hsien-te Kao, and Emilio Ferrara. Fair class balancing: Enhancing model fairness without observing sensitive attributes. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 1715–1724, New York, NY, USA, 2020. Association for Computing Machinery.

[S14] Muhammad Usman, Divya Gopinath, Youcheng Sun, Yannic Noller, and Corina S Păsăreanu. Nn repair: constraint-based repair of neural network classifiers. In *Computer Aided Verification: 33rd International Conference, CAV 2021, Virtual Event, July 20–23, 2021, Proceedings, Part I 33*, pages 3–25. Springer, 2021.

[S15] Vincenzo Riccio and Paolo Tonella. Model-based exploration of the frontier of behaviours for deep learning system testing. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2020, page 876–888, New York, NY, USA, 2020. Association for Computing Machinery.

[S16] Rangeet Pan and Hridesh Rajan. On decomposing a deep neural network into modules. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2020, page 889–900, New York, NY, USA, 2020. Association for Computing Machinery.

[S17] Salah Ghamizi, Maxime Cordy, Martin Gubri, Mike Papadakis, Andrey Boystov, Yves Le Traon, and Anne Goujon. Search-based adversarial testing and improvement of constrained credit scoring systems. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2020, page 1089–1100, New York, NY, USA, 2020. Association for Computing Machinery.

[S18] Jinhan Kim, Jeongil Ju, Robert Feldt, and Shin Yoo. Reducing dnn labelling cost using surprise adequacy: An industrial case study for autonomous driving. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2020, page 1466–1476, New York, NY, USA, 2020. Association for Computing Machinery.

[S19] Max Hort, Jie M. Zhang, Federica Sarro, and Mark Harman. Fairea: A model behaviour mutation approach to benchmarking bias mitigation methods. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, page 994–1006, New York, NY, USA, 2021. Association for Computing Machinery.

[S20] Joymallya Chakraborty, Suvodeep Majumder, and Tim Menzies. Bias in machine learning software: Why? how? what to do? In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, page 429–440, New York, NY, USA, 2021. Association for Computing Machinery.

[S21] Simin Chen, Cong Liu, Mirazul Haque, Zihe Song, and Wei Yang. Nmtsloth: Understanding and testing efficiency degradation of neural machine translation systems. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2022, page 1148–1160, New York, NY, USA, 2022. Association for Computing Machinery.

[S22] Guanhong Tao, Weisong Sun, Tingxu Han, Chunrong Fang, and Xiangyu Zhang. Ruler: Discriminative and iterative adversarial training for deep neural network fairness. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2022, page 1173–1184, New York, NY, USA, 2022. Association for Computing Machinery.

[S23] Laurent Gomez., Marcus Wilhelm., José Márquez., and Patrick Duverger. Security for distributed deep neural networks: Towards data confidentiality & intellectual property protection. In *Proceedings of the 16th International Joint Conference on e-Business and Telecommunications - SECRYPT,*, pages 439–447. INSTICC, SciTePress, 2019.

[S24] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th International Conference on Software Engineering*, ICSE '18, page 303–314, New York, NY, USA, 2018. Association for Computing Machinery.

[S25] Ziqi Zhang, Yuanchun Li, Jindong Wang, Bingyan Liu, Ding Li, Yao Guo, Xiangqun Chen, and Yunxin Liu. Remos: reducing defect inheritance in transfer learning via relevant model slicing. In *Proceedings of the 44th International Conference on Software Engineering*, pages 1856–1868, 2022.

[S26] Mirazul Haque, Yaswanth Yadlapalli, Wei Yang, and Cong Liu. Ereba: Black-box energy testing of adaptive neural networks. In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, page 835–846, New York, NY, USA, 2022. Association for Computing Machinery.

[S27] Hazem Fahmy, Fabrizio Pastore, Mojtaba Bagherzadeh, and Lionel Briand. Supporting deep neural network safety analysis and retraining through heatmap-based unsupervised learning. *IEEE Transactions on Reliability*, 70(4):1641–1657, 2021.

[S28] Tejas S. Borkar and Lina J. Karam. Deepcorrect: Correcting dnn models against image distortions. *IEEE Trans-*

actions on Image Processing, 28(12):6022–6034, 2019.

[S29] Bing Yu, Hua Qi, Qing Guo, Felix Juefei-Xu, Xiaofei Xie, Lei Ma, and Jianjun Zhao. Deeprepair: Style-guided repairing for deep neural networks in the real-world operational environment. *IEEE Transactions on Reliability*, 71(4):1401–1416, 2022.

[S30] Hong-Linh Truong and Tri-Minh Nguyen. Qoa4ml - a framework for supporting contracts in machine learning services. In *2021 IEEE International Conference on Web Services (ICWS)*, pages 465–475, 2021.

[S31] Wenbo Guo, Dongliang Mu, Jun Xu, Purui Su, Gang Wang, and Xinyu Xing. Lemna: Explaining deep learning based security applications. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 364–379, New York, NY, USA, 2018. Association for Computing Machinery.

[S32] Xiaoning Du, Xiaofei Xie, Yi Li, Lei Ma, Yang Liu, and Jianjun Zhao. Deepstellar: Model-based quantitative analysis of stateful deep learning systems. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2019, page 477–487, New York, NY, USA, 2019. Association for Computing Machinery.

[S33] Yue Zhao, Hong Zhu, Kai Chen, and Shengzhi Zhang. Ai-lancet: Locating error-inducing neurons to optimize neural networks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 141–158, New York, NY, USA, 2021. Association for Computing Machinery.

[S34] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, page 1310–1321, New York, NY, USA, 2015. Association for Computing Machinery.

[S35] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: Automated whitebox testing of deep learning systems. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, page 1–18, New York, NY, USA, 2017. Association for Computing Machinery.

[S36] Joymallya Chakraborty, Suvodeep Majumder, Zhe Yu, and Tim Menzies. Fairway: A way to build fair ml software. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2020, page 654–665, New York, NY, USA, 2020. Association for Computing Machinery.

[S37] Zhenpeng Chen, Jie M. Zhang, Federica Sarro, and Mark Harman. Maat: A novel ensemble approach to addressing fairness and performance bugs for machine learning software. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2022, page 1122–1134, New York, NY, USA, 2022. Association for Computing Machinery.

[S38] Mengdi Zhang and Jun Sun. Adaptive fairness improvement based on causality analysis. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2022, page 6–17, New York, NY, USA, 2022. Association for Computing Machinery.

[S39] Quan Zhang, Yifeng Ding, Yongqiang Tian, Jianmin Guo, Min Yuan, and Yu Jiang. Advdoor: Adversarial backdoor attack of deep learning system. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2021, page 127–138, New York, NY, USA, 2021. Association for Computing Machinery.

[S40] Pei Huang, Yuting Yang, Minghao Liu, Fuqi Jia, Feifei Ma, and Jian Zhang. Weakened robustness of deep neural networks. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2022, page 126–138, New York, NY, USA, 2022. Association for Computing Machinery.

[S41] Pin Ji, Yang Feng, Jia Liu, Zhihong Zhao, and Zhenyu Chen. Asrtest: Automated testing for deep-neural-network-driven speech recognition systems. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2022, page 189–201, New York, NY, USA, 2022. Association for Computing Machinery.

[S42] Yu Li, Muxi Chen, and Qiang Xu. Hybridrepair: Towards annotation-efficient repair for deep learning models. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2022, page 227–238, New York, NY, USA, 2022. Association for Computing Machinery.

[S43] Jialai Wang, Han Qiu, Yi Rong, Hengkai Ye, Qi Li, Zongpeng Li, and Chao Zhang. Bet: Black-box efficient testing for convolutional neural networks. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2022, page 164–175, New York, NY, USA, 2022. Association for Computing Machinery.

[S44] Yingyi Zhang, Zan Wang, Jiajun Jiang, Hanmo You, and Junjie Chen. Toward improving the robustness of deep learning models via model transformation. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, ASE '22, New York, NY, USA, 2023. Association for Computing Machinery.

[S45] Jingyi Wang, Guoliang Dong, Jun Sun, Xinyu Wang, and Peixin Zhang. Adversarial sample detection for deep neural network through model mutation testing. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, pages 1245–1256, 2019.

[S46] Yuanchun Li, Jiayi Hua, Haoyu Wang, Chunyang Chen, and Yunxin Liu. Deeppayload: Black-box backdoor attack on deep learning models through neural payload injection. In *2021 IEEE/ACM 43rd International Conference*

on Software Engineering (ICSE), pages 263–274, 2021.

[S47] Jingyi Wang, Jialuo Chen, Youcheng Sun, Xingjun Ma, Dongxia Wang, Jun Sun, and Peng Cheng. Robot: Robustness-oriented testing for deep learning systems. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 300–311, 2021.

[S48] Jie M. Zhang and Mark Harman. ""ignorance and prejudice"" in software fairness. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 1436–1447, 2021.

[S49] Boyuan Chen, Mingzhi Wen, Yong Shi, Dayi Lin, Gopi Krishnan Rajbahadur, and Zhen Ming (Jack) Jiang. Towards training reproducible deep learning models. In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, page 2202–2214, New York, NY, USA, 2022. Association for Computing Machinery.

[S50] Xuanqi Gao, Juan Zhai, Shiqing Ma, Chao Shen, Yufei Chen, and Qian Wang. Fairneuron: Improving deep neural network fairness with adversary games on selective neurons. In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, page 921–933, New York, NY, USA, 2022. Association for Computing Machinery.

[S51] Stefanos Georgiou, Maria Kechagia, Tushar Sharma, Federica Sarro, and Ying Zou. Green ai: Do deep learning frameworks have different costs? In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, page 1082–1094, New York, NY, USA, 2022. Association for Computing Machinery.

[S52] Zixi Liu, Yang Feng, Yining Yin, and Zhenyu Chen. Deepstate: Selecting test suites to enhance the robustness of recurrent neural networks. In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, page 598–609, New York, NY, USA, 2022. Association for Computing Machinery.

[S53] Saeid Tizpaz-Niari, Ashish Kumar, Gang Tan, and Ashutosh Trivedi. Fairness-aware configuration of machine learning libraries. In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, page 909–920, New York, NY, USA, 2022. Association for Computing Machinery.

[S54] Rangeet Pan and Hridesh Rajan. Decomposing convolutional neural networks into reusable and replaceable modules. In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, page 524–535, New York, NY, USA, 2022. Association for Computing Machinery.

[S55] Xiyue Zhang, Xiaofei Xie, Lei Ma, Xiaoning Du, Qiang Hu, Yang Liu, Jianjun Zhao, and Meng Sun. Towards characterizing adversarial defects of deep learning software from the lens of uncertainty. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ICSE '20, page 739–751, New York, NY, USA, 2020. Association for Computing Machinery.

[S56] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[S57] Simin Chen, Zihe Song, Mirazul Haque, Cong Liu, and Wei Yang. Nicgslowdown: Evaluating the efficiency robustness of neural image caption generation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15365–15374, June 2022.

[S58] Ming Fan, Ziliang Si, Xiaofei Xie, Yang Liu, and Ting Liu. Text backdoor detection using an interpretable rnn abstract model. *IEEE Transactions on Information Forensics and Security*, 16:4117–4132, 2021.

[S59] Patrick Henriksen, Francesco Leofante, and Alessio Lomuscio. Repairing misclassifications in neural networks using limited data. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, pages 1031–1038, 2022.

[S60] Guoliang Dong, Jingyi Wang, Jun Sun, Yang Zhang, Xinyu Wang, Ting Dai, Jin Song Dong, and Xingen Wang. Towards interpreting recurrent neural networks through probabilistic abstraction. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*, pages 499–510, 2020.

[S61] Zi Peng, Jinqiu Yang, Tse-Hsun Chen, and Lei Ma. A first look at the integration of machine learning models in complex autonomous driving systems: a case study on apollo. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1240–1250, 2020.

[S62] Xinyu Gao, Yang Feng, Yining Yin, Zixi Liu, Zhenyu Chen, and Baowen Xu. Adaptive test selection for deep neural networks. In *Proceedings of the 44th International Conference on Software Engineering*, pages 73–85, 2022.

[S63] Renjue Li, Pengfei Yang, Cheng-Chao Huang, Youcheng Sun, Bai Xue, and Lijun Zhang. Towards practical robustness analysis for dnns based on pac-model learning. In *Proceedings of the 44th International Conference on Software Engineering*, pages 2189–2201, 2022.

[S64] Bing Sun, Jun Sun, Long H Pham, and Jie Shi. Causality-based neural network repair. In *Proceedings of the 44th International Conference on Software Engineering*, pages 338–349, 2022.

[S65] Michael Weiss and Paolo Tonella. Fail-safe execution of deep learning based systems through uncertainty monitoring. In *2021 14th IEEE conference on software testing, verification and validation (ICST)*, pages 24–35. IEEE, 2021.

[S66] Isaac Dunn, Hadrien Pouget, Daniel Kroening, and Tom Melham. Exposing previously undetectable faults in deep neural networks. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 56–66, 2021.

[S67] Anurag Dwarakanath, Manish Ahuja, Samarth Sikand, Raghotham M Rao, RP Jagadeesh Chandra Bose, Neville Dubash, and Sanjay Podder. Identifying implementation bugs in machine learning based image classifiers using metamorphic testing. In *Proceedings of the 27th ACM SIGSOFT international symposium on software testing and analysis*, pages 118–128, 2018.

[S68] Wei Ma, Mike Papadakis, Anestis Tsakmalis, Maxime Cordy, and Yves Le Traon. Test selection for deep learning systems. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 30(2):1–22, 2021.

[S69] Anjana Perera, Aldeida Aleti, Chakkrit Tantithamthavorn, Jirayus Jiarpakdee, Burak Turhan, Lisa Kuhn, and Katie Walker. Search-based fairness testing for regression-based machine learning systems. *Empirical Software Engineering*, 27(3):79, 2022.

[S70] Md Nafee Al Islam, Yihong Ma, Pedro Alarcon, Nitesh Chawla, and Jane Cleland-Huang. Resam: Requirements elicitation and specification for deep-learning anomaly models with applications to uav flight controllers. In *2022 IEEE 30th International Requirements Engineering Conference (RE)*, pages 153–165. IEEE, 2022.

[S71] Carl Wilhjelm and Awad A Younis. A threat analysis methodology for security requirements elicitation in machine learning based systems. In *2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pages 426–433. IEEE, 2020.

[S72] Jurgen Cito, Isil Dillig, Seohyun Kim, Vijayaraghavan Murali, and Satish Chandra. Explaining mispredictions of machine learning models using rule induction. In *Proceedings of the 29th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering*, pages 716–727, 2021.

[S73] Fuyuan Zhang, Sankalan Pal Chowdhury, and Maria Christakis. Deepsearch: A simple and effective blackbox attack for deep neural networks. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 800–812, 2020.

[S74] Shiqing Ma, Yingqi Liu, Wen-Chuan Lee, Xiangyu Zhang, and Ananth Grama. Mode: automated neural network model debugging via state differential analysis and input selection. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 175–186, 2018.

[S75] Xiaofei Xie, Lei Ma, Felix Juefei-Xu, Minhui Xue, Hongxu Chen, Yang Liu, Jianjun Zhao, Bo Li, Jianxiong Yin, and Simon See. Deephunter: a coverage-guided fuzz testing framework for deep neural networks. In *Proceedings of the 28th ACM SIGSOFT international symposium on software testing and analysis*, pages 146–157, 2019.

[S76] Haibin Zheng, Zhiqing Chen, Tianyu Du, Xuhong Zhang, Yao Cheng, Shouling Ji, Jingyi Wang, Yue Yu, and Jinyin Chen. Neuronfair: Interpretable white-box fairness testing through biased neuron identification. In *Proceedings of the 44th International Conference on Software Engineering*, pages 1519–1531, 2022.

[S77] Xuli Sun and Shiliang Sun. Adversarial robustness and attacks for multi-view deep models. *Engineering Applications of Artificial Intelligence*, 97:104085, 2021.

[S78] Chiappa Silvia, Jiang Ray, Stepleton Tom, Pacchiano Aldo, Jiang Heinrich, and Aslanides John. A general approach to fairness with optimal transport. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3633–3640, 2020.

[S79] Emily Black, Samuel Yeom, and Matt Fredrikson. Fliptest: fairness testing via optimal transport. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 111–121, 2020.

[S80] Baolin Zheng, Peipei Jiang, Qian Wang, Qi Li, Chao Shen, Cong Wang, Yunjie Ge, Qingyang Teng, and Shenyi Zhang. Black-box adversarial attacks on commercial speech platforms with minimal information. In *Proceedings of the 2021 ACM SIGSAC conference on computer and communications security*, pages 86–107, 2021.

[S81] Ying Li, Yifan Sun, and Adwait Jog. Path forward beyond simulators: Fast and accurate gpu execution time prediction for dnn workloads. In *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 380–394, 2023.

[S82] Hazem Fahmy, Fabrizio Pastore, Lionel Briand, and Thomas Stifter. Simulator-based explanation and debugging of hazard-triggering events in dnn-based safety-critical systems. *ACM Transactions on Software Engineering and Methodology*, 32(4):1–47, 2023.

[S83] Zhenpeng Chen, Jie M Zhang, Federica Sarro, and Mark Harman. A comprehensive empirical study of bias mitigation methods for machine learning classifiers. *ACM Transactions on Software Engineering and Methodology*, 32(4):1–30, 2023.

[S84] Zhenya Zhang, Deyun Lyu, Paolo Arcaini, Lei Ma, Ichiro Hasuo, and Jianjun Zhao. Falsifai: Falsification of ai-enabled hybrid control systems guided by time-aware coverage criteria. *IEEE Transactions on Software Engineering*, 49(4):1842–1859, 2022.

[S85] Shikai Guo, Haorui Lin, Jiaoru Zhao, Hui Li, Rong Chen, Xiaochen Li, and He Jiang. An easy data augmentation approach for application reviews event inference. *IEEE Transactions on Software Engineering*, 2023.

[S86] Tao Zhang, Tianqing Zhu, Kun Gao, Wanlei Zhou, and S Yu Philip. Balancing learning model privacy, fairness, and accuracy with early stopping criteria. *IEEE Transactions on Neural Networks and Learning Systems*, 34(9):5557–5569, 2021.

[S87] Joanna Isabelle Olszewska. Snakes in trees: An explainable artificial intelligence approach for automatic object detection and recognition. In *ICAART (3)*, pages 996–1002, 2022.

[S88] Qiang Hu, Yuejun Guo, Xiaofei Xie, Maxime Cordy, Mike Papadakis, Lei Ma, and Yves Le Traon. Aries: Efficient testing of deep neural networks via labeling-free accuracy estimation. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 1776–1787. IEEE, 2023.

[S89] Shangqing Liu, Bozhi Wu, Xiaofei Xie, Guozhu Meng, and Yang Liu. Contrabert: Enhancing code pre-trained models via contrastive learning. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 2476–2487. IEEE, 2023.

[S90] Sayem Mohammad Imtiaz, Fraol Batole, Astha Singh, Rangeet Pan, Breno Dantas Cruz, and Hridesh Rajan. Decomposing a recurrent neural network into modules for enabling reusability and replacement. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 1020–1032. IEEE, 2023.

[S91] Xiaoning Ren, Yun Lin, Yinxing Xue, Ruofan Liu, Jun Sun, Zhiyong Feng, and Jin Song Dong. Deeparc: Modularizing neural networks for the model maintenance. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 1008–1019. IEEE, 2023.

[S92] Antonio Mastropaolo, Luca Pascarella, Emanuela Guglielmi, Matteo Ciniselli, Simone Scalabrino, Rocco Oliveto, and Gabriele Bavota. On the robustness of code generation techniques: An empirical study on github copilot. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 2149–2160. IEEE, 2023.

[S93] Linyi Li, Yuhao Zhang, Luyao Ren, Yingfei Xiong, and Tao Xie. Reliability assurance for deep neural network architectures against numerical defects. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 1827–1839. IEEE, 2023.

[S94] Binhang Qi, Hailong Sun, Xiang Gao, Hongyu Zhang, Zhaotian Li, and Xudong Liu. Reusing deep neural network models through model re-engineering. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 983–994. IEEE, 2023.

[S95] Xinyu Gao, Zhijie Wang, Yang Feng, Lei Ma, Zhenyu Chen, and Baowen Xu. Benchmarking robustness of ai-enabled multi-sensor fusion systems: Challenges and opportunities. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 871–882, 2023.

[S96] Yuxuan Wan, Wenxuan Wang, Pinjia He, Jiazhen Gu, Haonan Bai, and Michael R Lyu. Biasasker: Measuring the bias in conversational ai system. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 515–527, 2023.

[S97] Yining Yin, Yang Feng, Shihao Weng, Zixi Liu, Yuan Yao, Yichi Zhang, Zhihong Zhao, and Zhenyu Chen. Dynamic data fault localization for deep neural networks. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1345–1357, 2023.

[S98] Fuyuan Zhang, Xinwen Hu, Lei Ma, and Jianjun Zhao. Deeprover: A query-efficient blackbox attack for deep neural networks. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1384–1394, 2023.

[S99] Xiaokai Wei, Sujan Kumar Gonugondla, Shiqi Wang, Wasi Ahmad, Baishakhi Ray, Haifeng Qian, Xiaopeng Li, Varun Kumar, Zijian Wang, Yuchen Tian, et al. Towards greener yet powerful code generation via quantization: An empirical study. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 224–236, 2023.

[S100] Alessandro Tundo, Marco Mobilio, Shashikant Ilager, Ivona Brandić, Ezio Bartocci, and Leonardo Mariani. An energy-aware approach to design self-adaptive ai-based applications on the edge. In *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 281–293. IEEE, 2023.

[S101] Nikolaos Louloudakis, Perry Gibson, José Cano, and Ajitha Rajan. Deltann: Assessing the impact of computational environment parameters on the performance of image recognition models. In *2023 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 414–424. IEEE, 2023.

[S102] Huizhong Guo, Jinfeng Li, Jingyi Wang, Xiangyu Liu, Dongxia Wang, Zehong Hu, Rong Zhang, and Hui Xue. Fairrec: Fairness testing for deep recommender systems. In *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 310–321, 2023.

[S103] Yisong Xiao, Aishan Liu, Tianlin Li, and Xianglong Liu. Latent imitator: Generating natural individual discriminatory instances for black-box fairness testing. In *Proceedings of the 32nd ACM SIGSOFT international symposium on software testing and analysis*, pages 829–841, 2023.

[S104] Zhenlan Ji, Pingchuan Ma, Shuai Wang, and Yanhui Li. Causality-aided trade-off analysis for machine learning fairness. In *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 371–383. IEEE, 2023.

[S105] Khan Mohammad Habibullah, Gregory Gay, and Jennifer Horkoff. Non-functional requirements for machine

learning: Understanding current use and challenges among practitioners. *Requirements Engineering*, 28(2):283–316, 2023.

[S106] Yujin Huang, Han Hu, and Chunyang Chen. Robustness of on-device models: Adversarial attack to deep learning models on android apps. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pages 101–110, 2021.

[S107] Sakshi Udeshi, Pryanshu Arora, and Sudipta Chattopadhyay. Automated directed fairness testing. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, ASE '18, page 98–108, New York, NY, USA, 2018. Association for Computing Machinery.

[S108] Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks. In *Computer Aided Verification: 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I 30*, pages 3–29. Springer, 2017.

[S109] Peixin Zhang, Jingyi Wang, Jun Sun, Guoliang Dong, Xinyu Wang, Xingen Wang, Jin Song Dong, and Ting Dai. White-box fairness testing through adversarial sampling. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ICSE '20, page 949–960, New York, NY, USA, 2020. Association for Computing Machinery.

[S110] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017.

[S111] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, 2018.

[S112] Yujin Huang and Chunyang Chen. Smart app attack: Hacking deep learning models in android apps. *IEEE Transactions on Information Forensics and Security*, 17:1827–1840, 2022.

[S113] Wei Huang, Youcheng Sun, Xingyu Zhao, James Sharp, Wenjie Ruan, Jie Meng, and Xiaowei Huang. Coverage-guided testing for recurrent neural networks. *IEEE Transactions on Reliability*, 71(3):1191–1206, 2022.

[S114] Aniya Aggarwal, Pranay Lohia, Seema Nagar, Kuntal Dey, and Diptikalyan Saha. Black box fairness testing of machine learning models. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2019, page 625–635, New York, NY, USA, 2019. Association for Computing Machinery.

[S115] Maxime Cordy, Steve Muller, Mike Papadakis, and Yves Le Traon. Search-based test and improvement of machine-learning-based anomaly detection systems. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2019, page 158–168, New York, NY, USA, 2019. Association for Computing Machinery.

[S116] Teodora Baluta, Zheng Leong Chua, Kuldeep S Meel, and Prateek Saxena. Scalable quantitative verification for deep neural networks. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 312–323. IEEE, 2021.

[S117] Jianmin Guo, Yu Jiang, Yue Zhao, Quan Chen, and Jiaguang Sun. Dlfuzz: Differential fuzzing testing of deep learning systems. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 739–743, 2018.

[S118] Jiayang Song, Deyun Lyu, Zhenya Zhang, Zhijie Wang, Tianyi Zhang, and Lei Ma. When cyber-physical systems meet ai: A benchmark, an evaluation, and a way forward. In *Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice*, pages 343–352, 2022.

[S119] Florian Tramèr, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, Jean-Pierre Hubaux, Mathias Humbert, Ari Juels, and Huang Lin. Fairtest: Discovering unwarranted associations in data-driven applications. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 401–416, 2017.

[S120] Michaela Hardt, Xiaoguang Chen, Xiaoyi Cheng, Michele Donini, Jason Gelman, Satish Gollaprolu, John He, Pedro Larroy, Xinyu Liu, Nick McCarthy, Ashish Rathi, Scott Rees, Ankit Siva, ErhYuan Tsai, Keerthan Vasist, Pinar Yilmaz, Muhammad Bilal Zafar, Sanjiv Das, Kevin Haas, Tyler Hill, and Krishnaram Kenthapadi. Amazon sagemaker clarify: Machine learning bias detection and explainability in the cloud. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 2974–2983, New York, NY, USA, 2021. Association for Computing Machinery.

[S121] David Nigenda, Zohar Karnin, Muhammad Bilal Zafar, Raghu Ramesha, Alan Tan, Michele Donini, and Krishnaram Kenthapadi. Amazon sagemaker model monitor: A system for real-time insights into deployed machine learning models. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 3671–3681, New York, NY, USA, 2022. Association for Computing Machinery.

[S122] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In Sorelle A. Friedler and Christo Wilson, editors, *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 77–91. PMLR, 23–24 Feb 2018.

[S123] Sumon Biswas and Hridesh Rajan. Fair preprocessing: Towards understanding compositional fairness of data

transformers in machine learning pipeline. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, page 981–993, New York, NY, USA, 2021. Association for Computing Machinery.

[S124] Yutao Hu, Suyuan Wang, Wenke Li, Junru Peng, Yueming Wu, Deqing Zou, and Hai Jin. Interpreters for gnn-based vulnerability detection: Are we there yet? In *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 1407–1419, 2023.

[S125] Mathias Schneider, Ruben Prokscha, Seifeddine Saadani, and Alfred H¨oß. Ecba-mli: Edge computing benchmark architecture for machine learning inference. In *2022 IEEE International Conference on Edge Computing and Communications (EDGE)*, pages 23–32. IEEE, 2022.

[S126] Junming Cao, Bihuan Chen, Longjie Hu, Jie Gao, Kaifeng Huang, Xuezhi Song, and Xin Peng. Characterizing the complexity and its impact on testing in ml-enabled systems: A case sutdy on rasa. In *2023 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 258–270. IEEE, 2023.

[S127] Andria Procopiou and Andriani Piki. The 12th player: Explainable artificial intelligence (xai) in football: Conceptualisation, applications, challenges and future directions. In *Proceedings of the 11th International Conference on Sport Sciences Research and Technology Support*, volume 1, pages 213–220. Science and Technology Publications, Lda, 2023.

[S128] Matteo Rizzo, Alberto Veneri, Andrea Albarelli, Claudio Lucchese, Marco Nobile, and Cristina Conati. A theoretical framework for ai models explainability with application in biomedicine. In *2023 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–9. IEEE, 2023.

[S129] Luiz Marcio Cysneiros, Majid Raffi, and Julio Cesar Sampaio do Prado Leite. Software transparency as a key requirement for self-driving cars. In *2018 IEEE 26th international requirements engineering conference (RE)*, pages 382–387. IEEE, 2018.

[S130] Samarth Sikand, Vibhu Saujanya Sharma, Vikrant Kaulgud, and Sanjay Podder. Green ai quotient: Assessing greenness of ai-based software and the way forward. In *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 1828–1833. IEEE, 2023.