Classification, Challenges, and Automated Approaches to Handle Non-Functional Requirements in ML-Enabled Systems: A Systematic Literature Review

VINCENZO DE MARTINO, Software Engineering (SeSa) Lab, University of Salerno, Italy
FABIO PALOMBA, Software Engineering (SeSa) Lab, University of Salerno, Italy

Machine learning (ML) is nowadays so pervasive and diffused that virtually no application can avoid its use. Nonetheless, its enormous potential is constantly threatened by non-functional requirements, such as sustainability. In particular, we noticed the lack of a comprehensive synthesis of the research efforts done so far and how these may drive further research. In this paper, we propose a systematic literature review targeting three key aspects such as (1) the classification of the non-functional requirements investigated so far, (2) the challenges to face when dealing with them, and (3) the automated approaches proposed in literature to support practitioners when optimizing them in practice. Through the combination of well-established guidelines for conducting systematic literature reviews and additional search criteria, we survey a total amount of 69 research articles. Our findings report that current research identified 30 different non-functional requirements, which can be grouped into six main classes. We also deliver a catalog of over 23 software engineering challenges that further research should consider, besides an overview of the automated approaches researchers proposed to support practitioners when optimizing non-functional requirements of machine learning-enabled systems. We conclude our work by distilling implications and a future outlook on the topic.

CCS Concepts: • **Computing methodologies → Machine learning**; • **Software and its engineering → Extra-functional properties**.

Additional Key Words and Phrases: Software Engineering for Artificial Intelligence, Non-Functional Requirements, Systematic Literature Reviews.

ACM Reference Format:
Vincenzo De Martino and Fabio Palomba. 2023. Classification, Challenges, and Automated Approaches to Handle Non-Functional Requirements in ML-Enabled Systems: A Systematic Literature Review. 1, 1 (December 2023), 56 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Machine learning (ML) is now, more than ever, being used in theory, experiment, and simulation [71, 75]. On the one hand, companies and individuals increasingly rely on the outcome of machine learning models to make informed decisions [107] or automate tasks that would take substantial human workload [74]. 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 [64, 70]. As such, the pervasiveness of machine learning-intensive systems is expected to increase further in the coming years in multiple domains [73, 105].

Authors' addresses: Vincenzo De Martino, vdemartino@unisa.it, Software Engineering (SeSa) Lab, University of Salerno, Italy; Fabio Palomba, fpalomba@unisa.it, Software Engineering (SeSa) Lab, University of Salerno, Italy.

XXXX-XXXX/2023/12-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

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 [36, 69]. In particular, a non-functional attribute is defined as a condition that specifies a criterion that may be used to judge the operation of a system rather than specific behaviors [16]. In the context of machine learning-enabled systems, non-function attributes may affect the overall level of reliability, trustworthiness, and sustainability of these systems [33, 37, 55]. 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 machine learning-intensive systems, other than proposing methods and instruments to support practitioners when dealing with them [12, 32, 41]. 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 so-called *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 [17, 27, 103], security [34, 53], privacy [51, 81], and more [46]. 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 with respect to the management of non-functional requirements of ML-intensive systems, there is still a lack of a comprehensive, systematic synthesis of the current knowledge on the* ***non-functional attributes affecting machine learning-enabled systems****, the* ***challenges faced when dealing with them****, and the* ***automated approaches proposed to support practitioners when optimizing them throughout the software development lifecycle****.*

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 machine learning-intensive software systems. Our work follows well-established guidelines [43, 94] and additional search criteria based on seed set identification [66] to comprehensively synthesize existing research. From an initial set composed of over 2,500 hits, and after applying multiple snowballing rounds and additional data collection procedures, we ended up analyzing more than 81,000 research results. Through the application of exclusion/inclusion criteria and a rigorous quality assessment, we finally selected 69 papers. In addition, we provide a novel catalog composed of more than 23 software engineering challenges to deal with them, other than an overview of the automated approaches proposed by researchers to support practitioners when optimizing non-functional requirements throughout the software lifecycle. 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.

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

Table 1. Comparison with the Closest Related Work.

Related Work	Main Focus	Commonalities and differences
Martinez-Fernandez et al. [59]	A systematic mapping review targeting all previous research on software engineering for artificial intelligence. The authors surveyed 248 studies, classifying the available research according to the SWEBOK areas [14].	• Similar research approach to the search, yet with some differences in terms of seed set identification; • A broader analysis of current research, with no specific focus on non-functional requirements; • While the work includes part of the research papers we analyze, it does not provide insights into the challenges, practices, and methods to deal with non-functional requirements of machine learning-intensive systems.
Habibullah et al. [32]	A non-systematic analysis of non-functional requirements of machine learning-intensive systems. The authors focused on (i) the identification of clusters of non-functional requirements, (ii) the estimation of the amount of relevant studies for a subset of non-functional requirements, and (iii) the definition of the scope of non-functional requirements.	• We approach the literature search with a systematic approach, hence making a more comprehensive analysis; • We did not limit ourselves to the estimation of the current interest in the matter but provided a systematic investigation; • We let additional factors arise, including challenges, practices, and methods to deal with non-functional requirements of machine learning-intensive systems.
Horkoff [37]	An experience report on the challenges that the Requirements Engineering (RE) research community is called to face when addressing non-functional attributes of machine learning-intensive systems.	• We approach the literature search with a systematic approach, hence making a more comprehensive analysis; • We let additional factors arise, including a comprehensive set of non-functional requirements for machine learning-intensive systems, other than practices and methods to deal with them; • We let the challenges to deal with non-functional requirements emerge from the scientific literature on the matter, hence approaching the research differently.
Habibullah and Horkoff [33]	An industrial, interview-based study aiming at assessing (i) the identification and measurement of non-functional requirements, (ii) the importance of non-functional requirements in the industry, and (iii) the challenges associated with the identified non-functional requirements.	• We let our findings emerge from the scientific literature on the matter, hence approaching the research differently; • We let additional factors arise, including a comprehensive set of non-functional requirements for machine learning-intensive systems, other than practices and methods to deal with them; • We did not limit ourselves to the observations made by the industry practitioners but provided a systematic investigation of the current relevance of non-functional requirements.
Ahmad et al. [6]	A systematic mapping study on the requirements specification and modeling approaches and tools.	• The work targets requirements engineering from the perspective of specification and modeling, while ours focuses on non-functional attributes; • The research method is similar, yet we perform a systematic literature review rather than a mapping study, hence being more restrictive in terms of quality assessment of the primary studies;

Structure of the paper. Section 2 analyzes the related work. Section 3 describes the research goals and the methods applied to address them, while Section 4 discusses the results achieved. The implications and actionable insights are elaborated in Section 5, with the limitations reported in Section 6. Finally, Section 7 concludes the paper and outlines our future research agenda.

2 RELATED WORK AND MOTIVATION

To the best of our knowledge, no systematic literature review has been conducted with the aim of classifying non-functional requirements of machine learning-enabled systems and summarizing the challenges and automated approaches to deal with them. 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 [59], (2) explore, in a preliminary fashion, the relevance and research interest around non-functional requirements of machine learning-intensive systems

[32], and (3) summarize some of the key academic and industrial challenges when dealing with non-functional requirements in industry [33, 37]. These works are clearly the most closely related to ours. This section describes the major differences and limitations of current literature that motivated our research. A summary is provided in Table 1.

First and foremost, Martinez-Fernandez et al. [59] 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 for 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 and automated approaches proposed to deal with them. 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. [32] recently investigated the topic of non-functional requirements of machine learning-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. [32] share the same overall objective of our paper, i.e., an improved understanding of non-functional requirements of machine learning-intensive systems. Nonetheless, we aimed to conduct a comprehensive, systematic literature review. At the same time, we focused on a larger set of research angles, including an investigation of the challenges and approached proposed to deal with non-functional requirements.

Horkoff [37] discussed the challenges that the requirements engineering research community would be called to face to address themes connected to non-functional requirements of machine learning-intensive systems. The author identified several challenges based on the extensive experience accumulated in the industry over the years. With respect to this paper, our systematic literature review aims to collect comprehensive pieces of information coming from the scientific community to provide insights into how researchers have addressed the problem so far. At the same time, it is worth saying that we could corroborate the relevance of some of the challenges identified by Horkoff [37] through the evidence coming from our work.

Habibullah and Horkoff [33] 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. On the one hand, we enlarge the knowledge of non-functional requirements in machine learning-enabled systems by synthesizing the current literature from various perspectives. On the other hand, we could corroborate and further extend the findings on the challenges identified through a different research method.

Finally, Ahmad et al. [6] recently performed a systematic mapping of the research on requirements engineering for artificial intelligence. The authors surveyed the approaches to specify requirements and the frameworks/tools/methods to model them. Our work is clearly complementary, as we focus on non-functional attributes, attempting to classify them and synthesize the current knowledge on the challenges and approaches to deal with them.

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 the artificial intelligence and software engineering research communities [90], other than on the software engineering challenges and solutions for developing artificial intelligence systems [30, 46, 55, 67]. With respect to them, 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 [29, 61], design patterns [78, 92], software architecture [79], and testing [13, 15, 76, 104]. Finally, recent systematic literature reviews have been conducted with the aim of understanding the deployment strategies for artificial intelligence systems [40], and model-based development approaches [54].

The substantial efforts put by researchers in conducting systematic investigations on software engineering for artificial intelligence further motivate our work, as they let arise the collective, emerging interest around these matters. Our systematic literature review aims, therefore, to provide additional insights to the research community by studying the non-functional requirements of machine learning-enabled systems that have been studied over the last years.

3 RESEARCH METHOD

The ultimate *goal* of the study was to systematically survey the current research on non-functional requirements of machine learning-enabled systems, with the *purpose* of providing researchers in the field of SE4AI with actionable items and insights that they can exploit to further explore the matter and improve the support provided to machine learning engineers. Through our work, we therefore provided a *synthesis* of the state of the art rather than a *comparison* between the state of the art and the state of the practice: such a comparison would be undoubtedly beneficial, yet is deemed as part of our future research agenda. Specifically, the research was conducted in terms of (1) the classification of the non-functional requirements of ML-enabled systems, (2) the challenges that non-functional requirements pose during the development of ML-enabled systems, and (3) the automated approaches that may support practitioners throughout the development lifecycle. The *perspective* is of fresh Ph.D. students, senior researchers, and practitioners. The former are interested in mapping the research literature that focuses on addressing non-functional requirements in machine learning-enabled systems so that they can possibly identify neglected research areas to further investigate. Senior researchers are interested in mapping the state of the art in order to identify opportunities for research proposals and industrial collaborations. Finally, practitioners are interested in understanding the challenges to face when optimizing non-functional requirements during the development of ML-enabled systems, other than the current support provided by researchers, hence possibly identifying chances for technological transfer of those approaches in the industry.

To address our goal, we designed and executed a systematic literature review (SLR), which is a process through which the existing scientific papers on a subject of interest are rigorously identified, selected, and analyzed to address one or more research questions [43]. To make our SLR as complete as possible, we first followed the well-established guidelines proposed by Kitchenham et al. [42]. However, as further elaborated in Section 3.2, we had to preserve the sustainability of the data collection and analysis procedures when designing the search string. Specifically, we defined a search string that looked for articles that explicitly referred to terms like *"Non-Functional*

Requirements", possibly missing the articles that referred to the specific non-functional requirements considered, e.g., *"fairness"*. To mitigate threats due to incompleteness of the search, we boosted the search process by means of two additional steps. On the one hand, we systematically screened the research articles published in top-tier software engineering and artificial intelligence venues in an effort to identify a set of seed papers to further process, i.e., the identification of seed papers has been shown to be an effective alternative when investigating research angles for which the standard guidelines do not enable a comprehensive search [96]. On the other hand, we provided additional rigor to the analysis by integrating the so-called snowballing procedure [95], i.e., the iterative scanning of the incoming and outcoming references of the primary studies done to identify additional relevant sources of information. Therefore, our systematic literature review can be considered *"hybrid"* [66], namely a systematic analysis that 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.

3.1 Research Objectives and Questions

The Goal-Question-Metric (GQM) approach [18] was initially employed to purposefully measure the goals to be achieved and relate them to the data analyzed. More specifically, the overall research objectives established by this systematic literature review were the following:

> ⊙ ***Objective 1.*** *Providing a systematic classification of the non-functional requirements of ML-enabled software defined in the literature.*
> ⊙ ***Objective 2.*** *Providing a systematic classification of the challenges to face when dealing with non-functional requirements of ML-enabled systems.*
> ⊙ ***Objective 3.*** *Synthesizing the automated approaches proposed to handle non-functional requirements in ML-enabled software throughout the development lifecycle.*

Starting from these objectives, we defined more specific research questions (**RQ**s) that could drive our search process and analysis. These are presented in Table 2.

Table 2. Research Questions of the Systematic Literature Review.

Research Question	Description
RQ$_1$	What are the non-functional requirements of ML-enabled software considered by researchers?
RQ$_2$	What are the challenges of dealing with non-functional requirements of ML-enabled software?
RQ$_3$	What automated approaches are proposed to deal with non-functional requirements of ML-enabled systems?

More particularly, **RQ**$_1$ addressed the first objective of the study and aimed at systematically classifying the non-functional requirements that researchers have defined. By addressing this research angle, we aimed to contribute to previous research on the classification of non-functional

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

requirements of ML-enabled systems [32]. Indeed, while previous efforts in this respect have been conducted, there is still not a systematic classification - this represents a key contribution to our work. The second objective was mapped onto $\mathbf{RQ}_2$. This research question elicited the major challenges that non-functional requirements pose during the development of ML-enabled systems. On the one hand, this research question allowed us to understand the rationale behind the existence of the automated approaches proposed in the literature. On the other hand, $\mathbf{RQ}_2$ was able to enlarge the body of knowledge on the challenges to face when dealing with non-functional requirements, possibly informing researchers and practitioners on the aspects to further consider during the development of these systems. Finally, the third objective was mapped onto $\mathbf{RQ}_3$. The final step of our systematic literature review aimed at classifying which are the existing automated approaches, in an effort to provide an overview of the current status of the research on the matter. First, the results of this research question may inform researchers about neglected research areas to investigate. Second, $\mathbf{RQ}_3$ may be useful to practitioners to overview the currently available support, possibly finding technological transfer opportunities.

3.2 Research Method to Conduct the Systematic Literature Search

As a first step to address the goals of our study, we applied the guidelines by Kitchenham et al. [43] to identify primary studies targeting non-functional requirements of ML-enabled systems.

3.2.1 Research Query definition. To develop an effective search string, we first extracted relevant terms from the research questions, identifying the keywords from the **RQs** [43]. For all the terms, we then elaborated on the alternative spellings and synonyms. 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. More specifically, 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*”))*

There are some considerations to make on the definition of the search string. In the first place, it 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.

In the second place, 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 been ensured the collection of a larger amount of relevant papers. Nonetheless, we would have faced two issues. First, we would have been bound to the inclusion of the non-functional requirements explicitly classified in previous studies while defining the search string: as a consequence, we could not have satisfactorily addressed $\mathbf{RQ}_1$, not being able to classify additional non-functional requirements emerging from primary studies. Perhaps more importantly, the systematic literature review would have become prohibitive in terms of effort. For instance, Habibullah et al. [33] estimated the number of hits for a search query that includes all classified non-functional requirements in over 200,000 articles: on the one hand, such a search query would have had a low precision, identifying a high amount of irrelevant resources; on the other hand, the application of exclusion and inclusion criteria over such a large set of candidate articles would have required an excessive effort. As such, we had to identify an alternative solution, opting for the implementation of a hybrid mechanism to search the additional relevant resources through the processes described in Sections 3.3 and 3.4.

In the third place, we did not include terms connected to the development of automated approaches. We made this choice on purpose, as the first two research questions were not strictly connected to the automated approaches proposed in the literature but aimed at providing an overview of the non-functional requirements defined and the challenges they pose to the development of ML-enabled systems. On a similar note, we did not include terms like *"data mining"*, *"software engineering"*, *"software analytics"*, which would have restricted the scope of the search.

Data mining refers to the set of techniques that may extract data [5]. On the one hand, its inclusion would have possibly extended the data collection to the scientific literature not related to the development of ML-enabled systems or not in the scope of our work, e.g., a paper discussing a data mining technique for extracting non-functional requirements of traditional software. On the other hand, the keyword might have had a side effect, narrowing too much the scope of the systematic literature review: it would have indeed possibly limited the analyses to the non-functional requirements and automated approaches that can be employed at the time of data mining and engineering, hence not considering those that can be applied through a post-training operation.

The use of the term *"software engineering"* would have restricted the scope of the search to the papers that explicitly mentioned both *"machine learning"* and *"software engineering"*. This would have significantly reduced the effort required for the data collection and analysis. At the same time, we might have missed papers that were in scope but that did not refer to software engineering. This was a concern in our case, as some of the collected papers came from AI venues that do not use the term *"software engineering"* but mentioned terms like *"non-functional requirement"*. A similar discussion might be done for the term *"software analytics"*. By including it in the search string, we might have reduced the scope of the work to the papers that explicitly reported the use of software analytics instruments to handle non-functional requirements of ML-enabled systems. There were two potential issues here. On the one hand, we might have been too restrictive, not allowing the inclusion of articles published in venues different from those of software engineering. On the other hand, the articles collected from the execution of the search string should have addressed all the research questions of the work: two of them were not related to the automated approaches but rather aimed at providing a taxonomy of non-functional requirements of ML-enabled systems and of the challenges faced when addressing them: as such, we might have missed the analysis of relevant papers required to address $\mathbf{RQ}_1$ and $\mathbf{RQ}_2$.

More in general, we did not include keywords that might have led to the analysis of the broad field of research in artificial intelligence for software engineering (AI4SE) [63, 89, 93]. This was done on purpose, as our goal is exactly the opposite, namely that of contributing to the field of software engineering for artificial intelligence (SE4AI). More specifically, our work aims at contributing to the field of research that assesses how to build, maintain, and evolve ML-enabled systems rather than to the research field that explores the use of artificial intelligence techniques to support software engineering tasks. Our approach to the search string design is similar to the one of recent systematic literature reviews in the field of SE4AI proposed by Gezici and Tarhan [29] and Martinez-Fernandez et al. [59]: also in these cases, the authors did not include keywords that might have led to the analysis of scientific literature in the field of AI4SE.

Based on the arguments made above, we may conclude that we preferred to have a more generic search string in an effort to increase the recall of the search process—being able to collect a larger amount of papers—while preserving the sustainability of the overall data collection and analysis process. Of course, this decision impacted the precision of the search process and, subsequently, the effort required to apply the exclusion and inclusion criteria. Nonetheless, we accepted this compromise to make sure to include all relevant sources in our investigation.

3.2.2 Search Databases. As a second step of our investigation, we selected the databases to collect relevant resources. We applied the search on three well-known databases such as *ACM Digital Library*,[3] *Scopus*,[4] and *IEEEXplore*.[5] These digital libraries are often used to carry out systematic literature and mapping studies, as they provide 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 on the basis of the fact that 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. These databases only include peer-reviewed articles: as such, we did not need to double-check whether the papers extracted from them were peer-reviewed.

3.2.3 Exclusion and Inclusion criteria. To be considered useful for the goals of the systematic literature review, the papers retrieved from the search process had to be assessed against the following exclusion and inclusion criteria [42].

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

- Papers not written in English;
- Duplicated papers;
- Papers whose full-text read was not available;
- Paper not published or not peer-reviewed;
- Workshop, systematic, survey papers;
- Short papers, with a page count of less than five pages;
- Papers that did not fall into themes of computer science and computer engineering;
- Papers published before 2012;
- Conference papers which were later extended as journal submissions;
- Papers out of scope;

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

- Papers defining types of non-functional requirements of ML-enabled systems;
- Papers presenting approaches or tools to analyze non-functional requirements of ML-enabled systems;
- Papers presenting the domain in which non-functional requirements for ML-enabled systems are applied;
- Papers presenting methodologies to optimize non-functional requirements problems in ML-enabled systems.

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. [43] was found to be insufficient in finding relevant articles. Similarly to Habibullah et al. [33], we found out that the relevant papers might have referred to specific non-functional

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

requirements, e.g., *"fairness"*, rather than to the more generic term *"non-functional requirement"*. As such, the traditional systematic literature review approach might not have ensured completeness.

Table 3. 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 Pattern Analysis and Machine Intelligence* ***(TPAMI)****.*	**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 address this problem, we conducted the so-called *seed set identification*. This is the process through which a researcher systematically screens the research papers published in top conferences and journals over a given time period in an effort to identify additional resources relevant to the research they are conducting. Wohlin et al. [96] recently showed that this approach could be used to effectively complement a systematic literature review in cases where the standard guidelines cannot lead to a comprehensive search. In our context, the seed search process should have considered both software engineering and artificial intelligence venues, as studies targeting the analysis of non-functional requirements and/or the design of automated approaches to deal with them might have been published in both the research areas.

In the case of software engineering venues, we identified the most relevant software engineering conferences and journals, relying on (1) the *CORE* Ranking system[8] to select A* and A conferences, and (2) the *Scimago* Journal Ranking[9] to identify the software engineering journals falling into the first quartile of all journals (Q1) in the *'Software'* category. As such, we selected the conferences

[8]The *Computing Research and Education Association of Australasia*: https://www.core.edu.au/
[9]The *Scimago* Journal Ranking: https://www.scimagojr.com.

and journals marked as *'SE-related'* presented in Table 3. In the second place, we scanned all the papers published at those conferences starting from 2022 backward until 2012. As a final step, we analyzed each conference website, and for journals, we searched for papers via the DBLP,[10] 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).

In the case of artificial intelligence venues, opening up to all the most relevant ones would have been prohibitively expensive in terms of effort. Using the same criteria as for the software engineering venues, we would have selected 70 Q1 journals and more than 40 A* and A conferences, which we would have scanned from 2022 backward until 2012. We estimated the potential hits to analyze in around 200,000, which was deemed overly expensive for a human assessment. Nonetheless, to make our systematic literature process as complete as possible, we narrowed down the scope of the seed search to the artificial intelligence venues having a higher likelihood to publish engineering or empirical approaches to the development of ML-enabled systems, i.e., the venues that are more likely to contain papers relevant to our research questions.

In particular, we first identified an online repository, named *'AI Venues'*,[11], 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 decided to filter 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*. In this way, we could have focused on papers that had higher likelihood to be relevant for our research questions. Based on this process, we selected four Q1 journals and three A* conferences whose themes were either related to the improvement of AI approaches or the use of engineering or empirical approaches to AI. This process led to the selection of the venues marked as *'AI-related'* reported in Table 3. Also in this case, we scanned all the papers published to these venues between 2012 and 2022, applying the same selection process described in Section 3.2.3.

The seed search identified additional primary studies published to the selected venues. As a consequence, we did not need to double-check whether these were actually published or not.

3.4 Research Method to Conduct the Snowballing Process

The third step conducted to ensure the completeness of the search process revolved around the so-called *forward and backward snowballing*. This is the procedure through which a researcher systematically scans the incoming and outcoming references of the primary studies with the aim of identifying new relevant resources to address the research questions [94]. In our case, the primary studies identified as a consequence of the application of the exclusion and inclusion criteria on both the studies retrieved using the standard guidelines and the seed search were scanned. In particular, we applied an iterative procedure in which all incoming and outcoming 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

[10] The *DBLP*: https://dblp.org/.
[11] The *'AI Venues'* repository: https://aivenues.github.io/

were conducted through the use of *Google Scholar*,[12] an academic search engine which simplifies the analysis of incoming and outcoming references of research papers. This was the only step where we actually needed to verify whether the papers cited or citing the primary resources were published. Whenever needed, i.e., whenever we identified a new paper which 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. In the case 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. This step is recommended to ensure that the primary studies would have sound and reliable pieces of information to address the research questions of the study, while the papers that are not found to be of sufficient quality are discarded from the analysis [43].

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 in order to be useful to address our research questions. These were defined according to the research objectives initially defined:

- **Q1:** *Is the non-functional requirement well defined?*
- **Q2:** *If the primary study defined a challenge to deal with a non-functional requirement, is this well defined?*
- **Q3:** *If the primary study proposes an automated approach to deal with a non-functional requirement, are the data preparation steps well-defined?*
- **Q4:** *If the primary study proposes an automated technique to deal with a non-functional requirement, is the pipeline used to optimize it described?*

In particular, **Q1** was the instrument to ensure a proper answer to $\mathbf{RQ}_1$, while **Q2** mapped our requirements to address $\mathbf{RQ}_2$. Finally, **Q3** and **Q4** were relevant to understand the way automated approaches were fed and engineered and, as a consequence, to address $\mathbf{RQ}_3$.

When assessing the primary studies against each of the qualitative questions, previous systematic literature reviews (e.g., [6, 9, 22]) 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* [7], 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:

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

- 0 => **No**
- 0.1-1.5 => **Rarely**
- 1.6-2.8 => **Partly**
- 2.9-3.6 => **Mostly**
- 3.7+ => **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 4 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 six 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.

Table 4. 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.
Terms concerning non-functional requirements	**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	**RQ$_1$**	This component is similar to row 2 but contains all the domains in which non-functional requirements have been studied and found to be problematic. This will make it possible to diversify the impact of AI according to context and to identify the least researched areas.	ML Domains Environmental Domains
Challenges of SE approaches for ML-enabled systems	**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
Data-driven terms	**RQ$_3$**	The term data-driven includes all the techniques and technologies used on the datasets before training the models and the ML activities for each dataset the researchers performed.	Datasets Non-functional requirement considered Technologies for testing and analyzing the datasets Pre-processing techniques
SE approaches for ML-enabled systems	**RQ$_3$**	This component contains the knowledge areas most suitable for SE approaches for ML-enabled systems. We will collect and classify SE approaches according to different domains of ML systems.	Non-functional requirement considered in each domain Architectures of models. Type of ML systems Tools, Frameworks, and Methodologies to optimize Non-Functional Requirements during and after training.

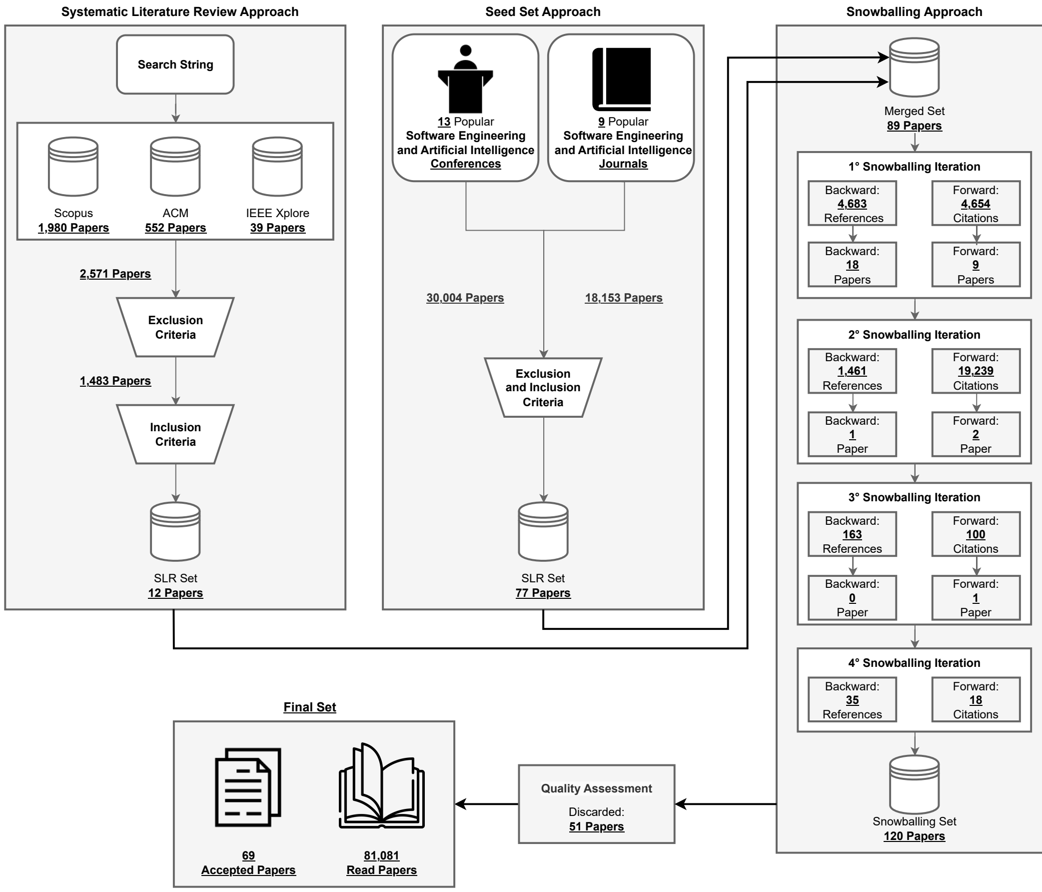

Fig. 1. Overview of the process for selecting papers.

3.7 Execution of the Research Methods

Figure 1 overviews the outputs of the execution of the research methods designed in the previous sections. The systematic literature review search was first conducted by the first author of the paper on September 4, 2023. As summarized in the figure, the application of the search query produced a total amount of 2,571 hits. As expected, most of these hits came from *Scopus* (1,980), with *ACM Digital Library* and *IEEEXplore* reporting a lower amount of results (552 and 39, respectively).

The first author of the paper manually collected all the relevant information on the resources identified and stored them within a sheet that was later shared with the second author. Afterwards, the first author proceeded with the application of exclusion and inclusion criteria: the former led to filtering out 1,088 research results, while the latter had a total of 1,471. The very limited amount of relevant papers identified after the application of the guidelines by Kitchenham et al. [43] led the two authors to open a discussion about the method defined and the implications that such an outcome might have had on the completeness of the search. In particular, the two authors scheduled two meetings of one hour each where they jointly reviewed the activities performed in an effort to understand the motivations behind the limited amount of resources retrieved. As a result of this fine-grained investigation, they could draw two main observations:

(1) The vast majority of the hits were found to be irrelevant to the goals of the study because they mentioned terms connected to non-functional requirements or machine learning either as part of the background or the related work, hence not discussing the challenges posed by non-functional requirements in the context of ML-enabled systems nor proposing automated approaches to cope with them;

(2) The few relevant hits which passed the inclusion criteria did not only explicitly mention the terms of our search string but also discussed the fine-grained requirements considered.

Based on these observations, the two authors realized the need for boosting the traditional search process with additional steps. While they first considered the re-execution of the systematic search through a search string designed to include all non-functional requirements classified by researchers so far, this would have led to three critical issues. First, the answer to $\mathbf{RQ}_1$ would have been biased by design: we would have indeed introduced ourselves to the set of non-functional requirements to look for rather than letting them emerge from the analysis of the available literature. Second, the search string would have included a number of *AND*/*OR* conditions that might have increased noise, leading to the retrieval of several resources not connected to SE4AI research. Last but not least, such an increased noise would have led to the identification of a prohibitively expensive amount of resources to analyze—Habibullah et al. [33] estimated over 200k hits for a search string including all the non-functional requirements of ML-enabled systems classified in literature.

To cope with these challenges, we opted for complementing the systematic literature search with the seed set identification and the snowballing process. Since our goal was that to analyze the literature available in the field of SE4AI on non-functional requirements, we then proceeded with the analysis of the papers published to top-tier software engineering and artificial intelligence venues. The first author of the paper looked for those resources, extracting information concerned to 30,004 papers from conferences and 18,153 papers from journals. After applying the exclusion and inclusion criteria, 77 unique new resources were deemed relevant, 71 from conferences and 6 from journals.

Before proceeding with the next steps, the second author of the paper reviewed the activities conducted so far. Also, two one-hour meetings were scheduled to discuss the results achieved. The meetings allowed us to acknowledge the soundness of the decision made when opting for a seed search. Hence, the 89 relevant resources acquired so far were taken as input for the snowballing step. Also, in this case, the first author was responsible for conducting the search process: by iterating multiple times, both backward and forward, the author was able to identify an additional set of 31 primary studies. Finally, the total amount of 120 papers identified through the three-step search process was considered within the scope of the quality assessment. Given the criticality of this step, this was jointly conducted by the authors: after analyzing 81,081 results, 69 primary studies finally contributed to our study.

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 800 person-hours; as for the second, it is about 400 person-hours.

Reproducibility and Replicability of Our Work.

Our online appendix reports all the fine-grained steps described above. The interested reader may use the appendix to replicate our work or build on top of our findings to create additional knowledge on non-functional requirements of ML-enabled systems: [60]

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.

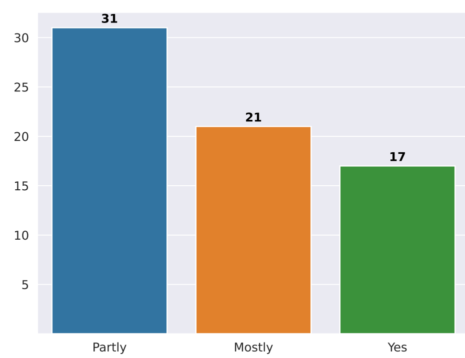

Fig. 2. Results of the quality assessment.

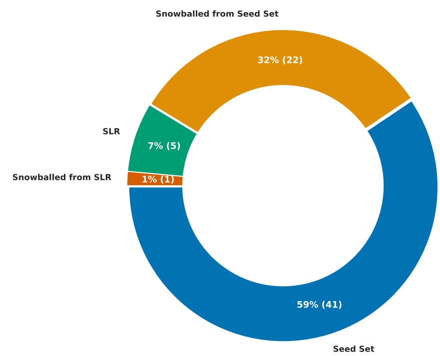

Fig. 3. Papers per publication venue.

4.1 Bibliometrics

Figure 2 overviews the results of the quality assessment procedure. We finally included a total of 69 primary studies. Most of them (31, 45%) received an overall quality score of *"Partly"*, i.e., they specified the pieces of information required to address our research questions mostly implicitly, yet giving us a chance to elicit them satisfactorily. The other 38 primary studies were more explicit and, indeed, reached an overall quality score of *"Mostly"* or *"Yes"*. The primary studies were mostly retrieved through the seed set search (41, 59%), while the other 22 resources (32%) were identified by snowballing the seed primary studies identified. Only 5 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.

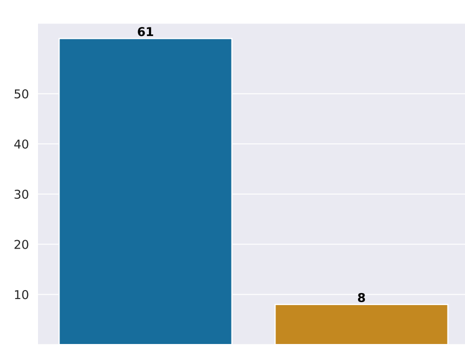

Fig. 4. Papers published per publication venue.

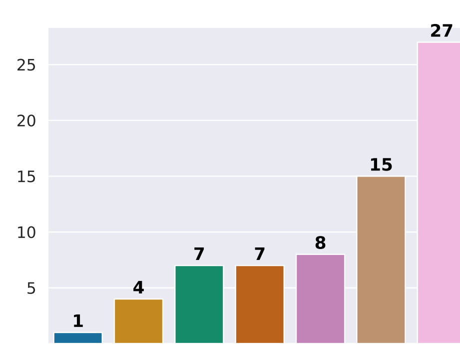

Fig. 5. Papers published on the matter over time.

More interesting were the insights coming from the publication venues, which are shown in Figure 4. We observed that the vast majority of the primary studies (61, 88%) appeared in conference proceedings, with a limited amount of resources published in journals (8, 12%). This finding seems to suggest that the research on non-functional requirements of ML-enabled systems is still at its early stage, with researchers interested in discussing recent advances in venues that allow discussion and interaction with the research community. On a similar note, the result may stimulate young researchers to enter this growing research area. The analysis of the amount of papers published on the matter per year confirmed the early nature of the research area (Figure 5). 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 two 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 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. The primary studies considered were first analyzed in an effort to elicit the non-functional attributes that they aimed to address. Afterward, we provided the extracted non-functional attributes with a (1) name, (2) a description, and (3) a reference class, i.e., a high-level class that may be used to cluster together multiple non-functional requirements based on similarity, i.e., non-functional requirements sharing similar purposes. For instance, requirements such as *"privacy"* and *"security"* were both extracted and mapped onto the *"Resiliency"* class, as they both referred to aspects making an ML-enabled system resilient to unexpected events arising in operation, including external attacks. 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; as such, they could be easily extracted and mapped onto known classes of non-functional requirements or onto new classes according to the conceptual similarity between the non-functional requirement analyzed and the set of classes identified. In the other cases, relevant terms connected to non-functional aspects were extracted, elaborated, and only later mapped onto the corresponding class of non-functional requirements. For example, the primary study [S1] reported terms like *"memory problems"* and *"battery drain"*, which we used to interpret the context of the study and assigned the non-functional aspect to the class entitled *"Sustainability"*. As an output of the process, we were able to develop a taxonomy of non-functional attributes of ML-enabled systems.

It is worth pointing out that 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* standard (ISO/EIC 25010). In particular, 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. The author exploited the existing body of knowledge in an effort to homogenize the definitions and classification; e.g., he favored the assignment of well-established class names whenever possible in an effort to provide a more comprehensive and usable taxonomy. The preliminary taxonomy was later the subject of evaluation. The second author joined the process at this stage, and a discussion was opened on the taxonomy produced. The two authors discussed about the consistency of the classification performed, other than on the names and descriptions assigned. Whenever needed, they

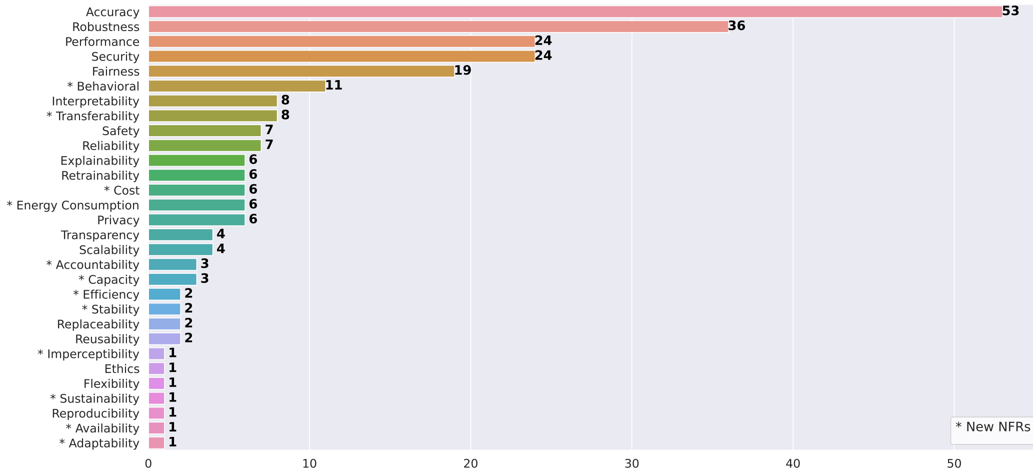

Fig. 6. Source-wise distribution of non-functional requirements in machine learning-enabled systems. A symbol '*' associated to one non-functional requirement denotes that it is a newly classified one.

modified the taxonomy—this happened in four cases, in which some non-functional requirements were grouped differently, and the names assigned to the classes were modified.

In the first place, as a result of this procedure we could identify a total amount of 30 non-functional requirements. These were: *accuracy* [S2–S54], *robustness* [S2–S7, S9, S10, S12, S19, S20, S22–S24, S27–S31, S40, S45, S46, S48, S49, S51, S53, S55–S64], *security* [S2, S3, S7, S9, S12, S14, S19, S22–S24, S27, S31–S33, S36, S45, S49, S55, S56, S58–S60, S62], *performance* [S1, S10, S13, S15–S17, S28–S30, S32, S37, S39, S40, S46, S50, S52, S53, S58–S60, S65], *fairness* [S13, S15, S17, S21, S26, S33–S35, S41, S42, S54, S61, S63–S69], *behavioral* [S7, S14, S20, S28, S43, S47–S49, S59, S61, S63], *interpretability* [S14, S17, S27, S33, S38, S39, S45, S59], *transferability* [S24, S31, S49, S55, S57, S58, S60], *safety* [S6, S7, S14, S24, S47, S50, S57], *reliability* [S20, S27, S28, S33, S48, S59, S62], *explainability* [S9, S21, S38, S39, S61, S68], *retrainability* [S3, S5, S23, S46, S62, S64], *cost* [S18, S22, S33, S37, S46], *energy consumption* [S1, S18, S37, S40, S51, S52], *privacy* [S8, S13, S32, S33, S36, S51, S62], *transparency* [S34, S39, S61, S67], *scalability* [S25, S27, S56, S57], *accountability* [S34, S66, S67], *Capacity* [S1, S37, S50], *efficiency* [S40, S53], *stability* [S27, S51], *replaceability* [S25, S44], *reusability* [S25, S44], *imperceptibility* [S6], *ethics* [S34], *flexibility* [S62], *sustainability* [S25], *reproducibility* [S16], and *availability*[S40], *adaptability* [S51].

Interestingly, our systematic exercise could identify not only non-functional requirements that emerged already in previous studies [32, 33, 37], 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*.

Figure 6 depicts the number of primary studies that targeted each of the 30 non-functional requirements identified. In the figure, the symbol '*' denotes the new non-functional requirements classified though our study, namely the ones that did not emerge from previous studies. As somehow expected, non-functional requirements such as *accuracy* and *robustness* of ML-enabled systems were those more frequently considered (53 and 36 times, respectively): these two aspects are likely the ones considered as recurrent and fundamental features of any machine learning model. Nonetheless, aspects connected to *performance* (24), *security* (24), *fairness* (19), and *behavioral (11)* seem to be growing research areas. On the contrary, multiple aspects were little explored, e.g., *reusability* or *scalability*, possibly indicating emerging research themes that would be worth further exploring.

Table 5. Non-Functional Requirements in Ml-enabled systems.

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

The classification exercise led to the identification of six main classes of non-functional requirements: Table 5 reports, for each class, the set of non-functional requirements belonging to the class, along with their definition. 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. 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., [32, 33, 37], 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 criterion that may be used to judge the operation of a system rather than specific behaviors"* [16]: 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 machine learning-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 machine learning-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.

Maintainability. Maintainability is chosen as a cluster to represent *replaceability*, *retrainability*, *reproducibility*, *transferability*, *reusability*, 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. By prioritizing maintainability, designers and developers can create reliable, efficient, adaptable, flexible, and sustainable systems over the long term.

Resiliency. Resilience was chosen as a cluster to represent *robustness*, *reliability*, *behavioral*, *flexibility*, *security*, *safety*, *privacy*, and *availability* because 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. By prioritizing resilience, designers and developers can build reliable and secure systems that can withstand disruptions and threats while maintaining their core functions and services.

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.[13] Researchers have been also arguing that fairness and ethical concerns should be considered as a form of sustainability. McGuire et al. [62] advocated that social sustainability refers to multiple dimensions, including pro-social vs. anti-social affordances. On a similar note, various other researchers [11, 24, 47] argued to consider social and ethical aspects as sustainability properties. Overall, sustainability can be

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

considered as a guiding principle for designing and developing efficient, cost-effective but also equitable, ethical, and accountable systems.

Usability. The usability class is similar to the one defined by Habibullah et al. [32]. 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.

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 machine learning-enabled systems, other than understanding the quality and efficiency aspects practitioners should focus on when releasing machine learning models.

4.2.1 On the non-functional requirement domains. Figure 7 overviews the number of primary studies targeting each machine learning domain extracted. As shown, the most frequent are *'Computer Vision'* (47), *'Decision Making'* (18), and *'Natural Language Processing'* (9), while other, less targeted domains pertain to emerging technologies, e.g., *'Recommender Systems'* or *'Automatic Speech Recognition'*. These insights raise some key ML domains that researchers have been analyzing in the recent past, but also raise contexts where further research might be worth focusing on. The much larger amount of primary studies targeting *'Computer Vision'* may indicate the critical nature of such a domain with respect to the management of non-functional requirements. This may be due to the nature of the inputs that the machine learning-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.

Besides the ML domains, we also analyzed the environments where the primary studies focused on. These are shown in Figure 8. The most common environment has been classified as *'General Environment'* (39) and indicates that most papers did not explicitly specify the environmental conditions which they experimented with. Other papers were instead more specific, targeting *'Driving Systems'* (12), *'Mobile Devices'* (7), and *'Critical/Evolving Scenarios'* (5). Looking at the

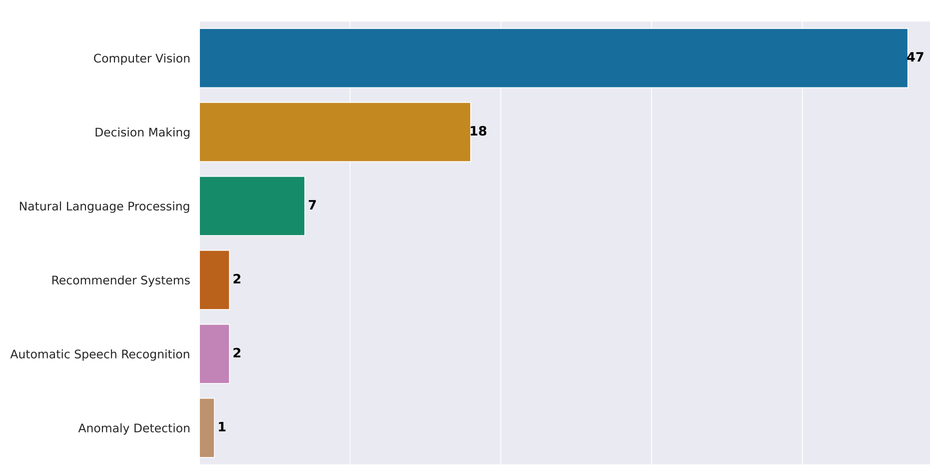

Fig. 7. Distribution of NFRs across ML domains in ML-enabled systems.

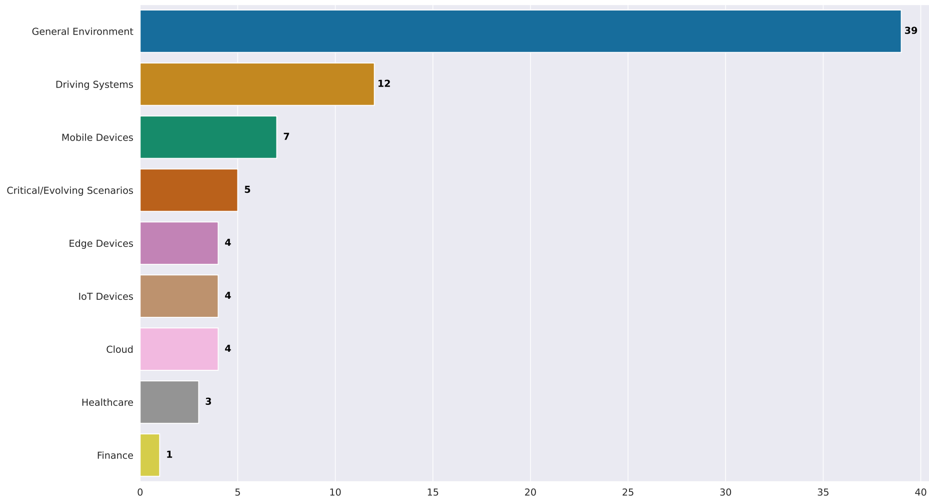

Fig. 8. Distribution of NFRs across Environment domains in ML-enabled systems.

figure, we may conclude that most primary studies worked in generalized conditions, with fewer researchers deepening the analysis of non-functional requirements in the context of ML-enabled systems designed to work under specific working conditions. This may represent a challenge for further research in the field: researchers might indeed be interested in further understanding the peculiarities of the various environments, possibly letting emerge novel non-functional requirements to deal with or learning how to prioritize non-functional requirements based on the environmental conditions that the ML-enabled systems are called to face.

Providing an additional perspective, Figure 9 illustrates a heatmap that shows the relation between the specific non-functional requirements presented in Table 5 and the corresponding environments depicted in Figure 8. Each entry in the heatmap represents a numerical value, indicating the frequency with which the 'i-th' non-functional requirement has been considered within the 'j-th' environment, with 'i' corresponding to the rows and 'j' to the columns. By looking at the columns, the heatmap shows which are the non-functional requirements often addressed within the same environment. By looking at the rows, the heatmap shows the environments where each non-functional requirement has been analyzed so far. Through this visualization, we could first observe that there exist a number of environments where the current knowledge seems to be limited: for example, the environments 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 environments. 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 9 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 environment considered. These insights may, again, be beneficial for researchers interested in working toward understanding non-functional requirements and their impact on ML-enabled

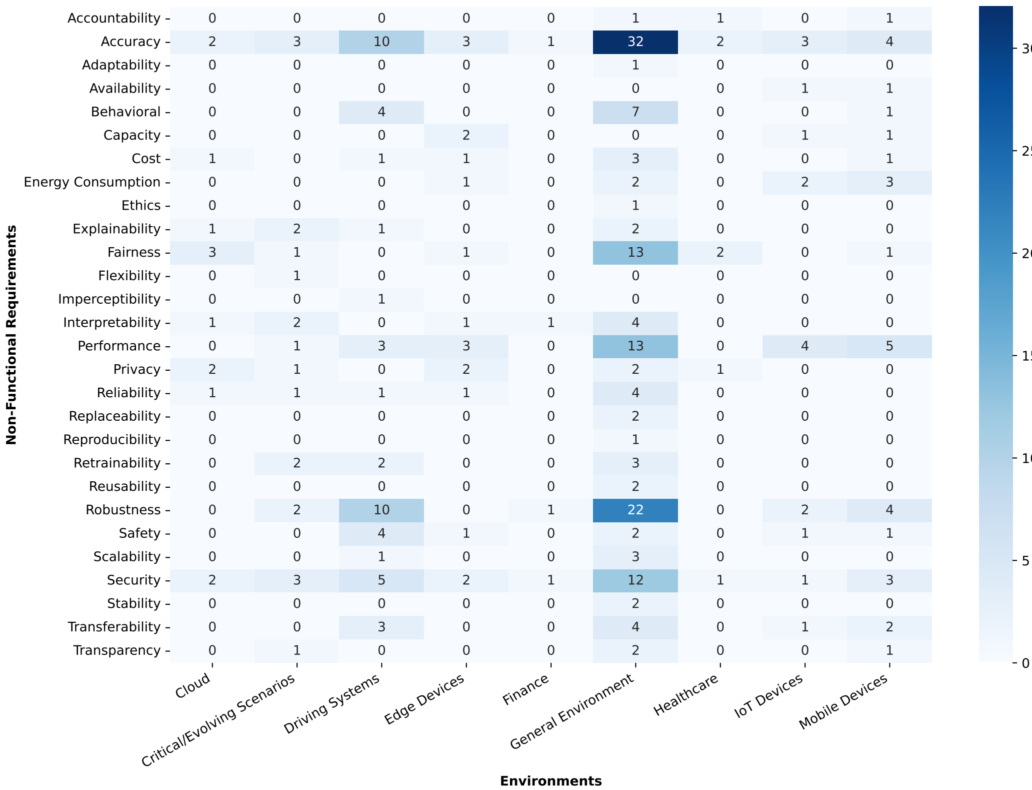

Fig. 9. Heatmap of Non-Functional Requirements across Environment domains in ML-enabled systems.

systems working in different environments, and their understanding may help make more informed decisions and improve system development practices.

Finally, Figure 9 also suggests the existence of possible trade-offs to further consider in future research. Indeed, while the figure does not indicate if more non-functional requirements were simultaneously addressed by researchers, it may still highlight potential relations. As an example, in the *'Driving Systems'* environment we noticed that *accuracy* and *robustness* were more frequently targeted, possibly indicating that these two non-functional requirements should jointly considered by researchers to enlarge the current body of knowledge on the trade-offs among non-functional requirements of ML-enabled systems.

Answer to $\mathbf{RQ}_1$. As a result of $\mathbf{RQ}_1$, we could identify a total of 30 relevant non-functional attributes, classified according to six main categories. We also identified the most frequent ML domains and environments 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 of dealing with non-functional requirements of ML-enabled software?

The second research question is concerned with the identification of the challenges faced in dealing with the non-functional requirements of machine learning-enabled software. Taking the primary studies as an input, we processed them individually. We first went through the content of each primary study in search of two pieces of information: (1) the 'context' where the non-functional requirement was considered; and (2) the 'problem(s)' the primary study aimed at addressing. The information about the context was helpful to elicit where a non-functional requirement should be more carefully taken into account, while the information about the problem considered by the primary study was used to elaborate on the motivations making a non-functional requirement hard to treat. After collecting these pieces of information, we then applied an inductive reasoning to elicit the challenge(s) associated with the management of non-functional requirements. This process was systematic and included (1) the identification of the issues addressed by the primary studies; (2) the thematic analysis of these issues; and (3) the classification of the challenges related to management of a non-functional requirement. As such, the set of challenges described in our work has been derived from the content of the primary studies through the analysis of the contexts and problems considered by these primary studies. In doing so, we focused on the software engineering perspective, hence attempting to produce open challenges that our research community is called to address through further investigations.

Our work identified more than twenty software engineering challenges for our research community. 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 [30]. In the second place, it is important to remark that accuracy represents a cross-cutting concern, i.e., all the challenges identified have implications for the accuracy of machine learning-enabled systems. For this reason, we preferred to discuss accuracy while presenting the challenges connected to the other classes of non-functional requirements identified in $\mathbf{RQ}_1$. The list of challenges is presented in the following.

Efficiency (E): In terms of overall performance and effectiveness of machine learning-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 Internal Errors.** Neural networks can produce erroneous outputs as a consequence of internal errors due to *incorrect parameters*, *incorrect weight values*, *uncovered root causes*, and *incorrect manual labeling*. Those internal errors may significantly impact the performance of machine learning-enabled systems, leading to reduced accuracy, slower inference times, and increased energy consumption [S1, S3, S37]. For example, incorrect weight values or parameters can cause the network to converge slowly or get stuck in suboptimal solutions, resulting in poor performance. Dealing with internal errors has two main connotations such as tracking errors in deep neural network models and addressing them. Our findings suggest that research in terms of *bug localization*, *performance monitoring*, and *program repair* might support the challenges identified.
- **Challenge E.2 - Dealing with High Latency.** Literature reported that multiple design choices related to *hyper-parameter tuning*, *model optimization*, and *model architecture* can affect the overall efficiency of neural networks, even though those decisions might sometimes lead to increased accuracy [S37]. In other terms, the definition of neural networks should be considered as an optimization problem, where designers look for the best trade-off between performance and other quality measures and carry out preliminary studies to find the best configurations to

employ within the specific use case. Additionally, the computational efficiency of DNN systems is very sensitive to even slightly different inputs. As a consequence, a change to the inputs may result in a significantly higher computational demand, deteriorating the overall models' efficiency [S40, S52, S53]. As such, researchers in the field of *software analytics*, *software architecture*, and *software quality and optimization* might help address this challenge by means of analytical and code quality instruments aiming at finding the right compromise between accuracy and efficiency.

- **Challenge E.3 - Dealing with Space.** A third critical challenge arising from our review consists of the space required by machine learning models and, particularly, by neural network-based solutions. These may indeed present memory issues due to their large size, which has been particularly relevant in contexts like edge computing and IoT. Deploying large models on such devices can result in memory constraints affecting system performance [S1, S37, S50]. In response to this challenge, designers should be able to apply policies and strategies to build reduced models before deploying them on devices. These strategies may mitigate memory issues and improve the overall performance of machine learning-enabled systems. In this respect, our findings call for further research on the emerging field of *tiny machine learning* [50], that is, the combination of hardware, algorithms, and software that support on-device sensor data analytics at low power. While some initial effort has been made in the artificial intelligence research community (e.g., [25, 80]), we point out a lack of knowledge of the software engineering side of the matter.

The three challenges identified were discussed in all the domains requiring the definition of large models, i.e., *'Computer Vision'*, *'Driving Systems'*, *'Natural Language Processing'*, *'Mobile Devices'* and *'IoT Devices'*. This was somehow expected, as scalability concerns become evident in environments requiring high computational demands.

Maintainability (M): When it turns to the ability of machine learning-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.** Machine learning-enabled systems based on deep learning models are known to be complex and difficult to reproduce accurately. One of the primary challenges in reproducing deep learning models concerns with 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 [S16]. Our findings seem, therefore, to suggest the need for mechanisms that could support the *verification* of deep learning models, for instance, by injecting elements of randomness and non-determinism to check how the system may behave in those circumstances.
- **Challenge M.2 - Increasing Model Decomposition and Reuse.** A critical challenge when building and improving machine learning-enabled systems based on deep learning models is the need to reuse parts of previously constructed models. This can be difficult, for instance, when attempting to replace potentially defective parts with others. A possible approach concerns with the identification of the parts responsible for each output class or module in the models: this would allow to selectively reuse or replace only the desired output classes to build the model. This challenge is strictly connected to the properties of randomness and non-determinism already discussed in the previous point. Indeed, increasing decomposition and reuse requires a proper understanding of the potential sources of variability and non-determinism in the system's software and hardware components. Therefore, improvements in terms of reproducibility may also lead to more reusable models [S16, S25, S44]. On the

one hand, our findings reinforce the need for research in terms of *verification*. On the other hand, the challenge of identifying reusable components seems to call for research in terms of *program slicing* and *refactoring*, which might support machine learning engineers to properly extract parts of the models for reusability purposes.

Challenges connected to maintainability were found to be pervasive in all domains, hence representing concerns that may impact any kind of machine learning-enabled system.

Resiliency (R): Improving the universal robustness and security of machine learning models present key issues [S12]. This was the class of non-functional requirements where we identified more challenges (8); 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.** Machine learning models, and more particularly deep neural networks, are vulnerable to the so-called *adversarial attacks* [57, 84], namely attacks that manipulate the input data in a way that causes the neural network to produce an incorrect output and significantly drop accuracy. Different types of adversarial attacks can be targeted or untargeted and can be carried out through various techniques such as *adversarial examples* [84], *adversarial perturbation* [10], *poisoned data* [38], and *backdoor samples* [31]. These types of attacks can negatively affect the overall robustness of the model and decrease its accuracy. As such, a key challenge is represented by the resilience to adversarial attacks [S2, S55, S58]. More specifically, existing literature highlighted the need for instruments able to design perturbations that may properly generate adversarial examples for different machine learning models, e.g., rotations, smoothing, and erosion operations in machine learning-enable systems targeting computer vision problems [S4, S6, S10, S57]. Those instruments might support practitioners in terms of an improved comprehension of how models can be affected by vulnerabilities, other than through improved methods to increase appropriate defenses against adversarial attacks. Hence, our findings suggest additional *verification and validation* research efforts: indeed, while the security perspective has been often targeted by the artificial intelligence community through the definition of new algorithms and techniques [49, 91], software engineering research might be complementary and propose novel verification and validation approaches, which may improve the policies and strategies employed by practitioners when testing for security.
- **Challenge R.2 - Exploiting the Sensibility of Adversarial Attacks.** A complementary perspective is concerned with the sensibility that certain adversarial attacks may have to input distortions and how such a sensibility can be exploited to improve the security of machine learning models. For example, in computer vision applications, attacks made through non-additive noise or geometric morphing may cause serious security concerns, yet those filters are sensitive to input distortions, meaning that automatic correction instruments may potentially distort the malicious inputs to mitigate their effects on the machine learning model [S10, S45, S55, S56]. Our findings, therefore, point out the need for techniques that may handle perturbations that not only occur naturally in the physical environment but could be maliciously generated by adversary attacks. The perturbations are typically stealthy and undetectable, which poses further challenges for systems based on deep neural networks [S4]: as such, characterizing them would be the first step for *empirical software engineering* and *automatic program repair* researchers.
- **Challenge R.3 - Security Verification of Pre-Trained Models.** Pre-trained models can be beneficial, as they reduce the computational burden of training complex deep learning models through transferability. However, pre-trained models may be vulnerable to attacks due to the

displacement of the dataset and the potential for malicious code or exploitation [S58, S60]. On the one hand, assessing the security risks of these models still represent a key challenge to face through the definition of software analytics instruments able to provide practitioners with security insights and best practices in selecting the most appropriate pre-trained model to use to avoid security concerns. On the other hand, existing literature [S19, S49, S58, S60] let also emerge additional challenges targeting the definition of design practices that may allow to develop and retrain security-aware fine-tuned models. Our findings, therefore, suggest that researchers in the field of *software analytics*, *software quality*, and *verification and validation* might contribute to addressing the challenges in this set.

- **Challenge R.4 - Resilience to Intellectual Property Theft.** Machine learning-enabled systems based on neural networks suffer from data breaches and unauthorized access to sensitive data due to training attacks [88], i.e., malicious attacks on training data. According to the existing literature, only a little knowledge is available on how to secure machine learning models against those types of attacks. Greater attention should be paid to strengthening security protocols [S32, S58]. While this mostly calls the attention of researchers in the field of networks and security, the software engineering research community might still contribute through the definition of more security best practices able to protect models from intellectual property theft, especially in edge devices and IoT systems [101].
- **Challenge R.5 - Diagnosing the Internal Behavior of Models.** Understanding the root cause of anomalous behaviors of deep neural networks and how to fix them represent critical aspects to further elaborate. On the one hand, evaluating the reliability implications of small sets of weights assigned to the network may support practitioners toward an improved understanding of how they work. In this respect, the definition of exploration, interpretability, and explainability methods may lead to substantial advances in terms of diagnosis of failures and misclassification, other than of security concerns [S9, S47, S48, S61]—Ji et al. [S28] also called for novel local models to uncover the root causes of deep neural network failures. On the other hand, existing literature also pointed out the need for novel assessment metrics that, besides accuracy, can provide actionable insights into the overall robustness of the model [S11, S27]. As such, our findings suggest multiple avenues for software engineering researchers in the field of *fault localization*, *explainable AI*, *software metrics*, and *automatic program repair*.
- **Challenge R.6 - Optimal Post-Deployment Simulation.** While some threats to resiliency might be managed at the development time, additional challenges arise at the deployment stage. In particular, machine learning models could not work as expected when deployed. The problem has been mainly pointed out in the context of automatic speech recognition [S28, S29], yet it may affect any kind of machine learning solution. The main challenge is related to properly understanding how the model would work in a real-case scenario. Besides software testing [76], literature identified post-deployment simulations as a complementary instrument that might be worth to further investigate: this refers to the definition of agent-based models that may (i) simulate the environment where the machine learning-enabled system would work and (ii) verify the system against a large variety of simulated inputs. This is, again, something that may catch the attention of researchers in the field of *verification and validation* and *machine learning for software engineering*.
- **Challenge R.7 - Preserving Privacy in Machine Learning-Enabled Systems.** The primary studies identified the challenge of developing privacy-preserving deep neural networks and machine learning systems, which may allow algorithms to run securely on distributed data without compromising sensitive information about the subjects of the data, leaving users with the ability to delete their data at any time leaving users with the ability to delete their data at

any time. This would require the definition of strategies to train models without sacrificing accuracy while ensuring the protection of private data [S8, S36]. As such, this challenge targets multiple software engineering fields, from *software quality* to *software architecture*, which may deepen the current knowledge on data encryption, anonymization techniques, federated learning, and differential privacy.

- **Challenge R.8 - Improving the Generalizability of Existing Automated Frameworks.** Multiple primary studies identified limitations and challenges related to the automated support provided by existing frameworks. Challenges in this category mainly concern with their generalizability. For instance, Plum [S5], a recommendation system to prioritize model repair strategies, is limited to the analysis of deep learning models and does not provide support for different learning tasks. Similarly, other approaches [S7, S20] can only assist practitioners when developing specific types of neural networks. As such, we also identified technological and empirical challenges for our research community, which is called to develop and experiment with automated solutions by keeping generalizability into account.

Challenges in this category were mainly considered in the domains of *'Computer Vision'*, *'Driving Systems'*, *'Mobile Devices'*, *'Edge Devices'*, and *'IoT Devices'*, i.e., those more intensively threatened by security issues. Nonetheless, we observed a rising interest toward *'Automatic Speech Recognition'*—this is likely due to the growing research on large language models [19].

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 seven 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.** Improving performance, accuracy, and fairness simultaneously is currently the main challenge for researchers [S13, S15, S41, S42]. More specifically, the challenge is to study and mitigate the impact of disparate results, offensive labeling, and uneven algorithmic error rates in data-driven applications. In addition, the primary studies investigated highlighted the relevance of dealing with algorithmic discrimination in the context of the pre-processing phase in an effort to improve the trade-off between fairness and performance in ML software. While the software engineering research community has already contributed to addressing this challenge (e.g., [S61, S66]), our findings suggest that further research in terms of *software analytics* and *verification and validation* might be worthwhile.
- **Challenge S.2 - Model Accountability:** 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. The primary studies considered in our work remarked that more effort should be given to this aspect to allow models to be deployed with an eye on ethics and fairness. In this respect, major challenges pertain to the definition of *verification and validation* instruments through which practitioners can verify how accountable their systems actually are [S66, S67]. For instance, decision making may represent ideal tools to assess the level of accountability of machine learning models.
- **Challenge S.3 - Fairness Analytics.** Another critical challenge for social sustainability is related to fairness analytics, that is, the definition of metric toolkits that may support practitioners in diagnosing fairness at the development time. However, fairness should not be considered as a non-functional requirement per se, but rather be part of trade-off analyses. More specifically, existing literature called for novel metrics able to provide insights into the

compromise between fairness and accuracy [S26], fairness and privacy [S8], and fairness and accountability [S66]. As such, the challenge is represented by the diagnosis and multi-objective optimization of data-driven applications in multiple development stages, from requirements engineering to verification and validation [S66].

- **Challenge S.4 - Improving Sustainability Benchmarks.** Current literature [S67] also highlighted the lack of low-quality of benchmark datasets that may assist the creation of sustainable machine-learning applications. These datasets should indeed be devised by enabling cross-domain and cross-sectional analysis. On the one hand, application domains focusing on the image processing would require high-quality datasets allowing automated pose, illumination, and expression (PIE) analysis. On the other hand, datasets presenting large and diverse information on minorities should be devised to allow the more effective training of machine learning systems.
- **Challenge S.5 - Reducing Energy Cost.** One of the most important challenges for deep neural networks is concerned with the reduction of energy and computational costs. Indeed, those systems have considerable energy and financial costs, which lead to increase CO2 emissions and memory consumption in both the training and post-release phases [S18, S20, S33, S37, S51, S52]. More specifically, energy consumption issues significantly impact edge and IoT devices, which indeed represent critical use cases. The software engineering research community is called to contribute in different manners. First, *empirical software engineering* research is called to assess the impact of different AI containerization strategies on energy consumption and memory utilization. Second, *software quality* and *software architecture* researchers might help address the trade-offs between training accuracy and post-deployment consumption. Last but not least, our findings call for further research on the definition of software engineering practices to develop *tiny machine learning* solutions.
- **Challenge S.6 - Increasing the Practitioner's Awareness of Sustainability.** Our literature work identified the challenge of increasing the developer's awareness with respect to sustainability concerns. This issue is particularly relevant when considering energy consumption, which is a critical concern in today's world. As such, the definition of novel strategies and instruments to make practitioners aware of the trade-offs between performance and sustainability represents a key challenge for our research community [S1, S18]. Once again, more research on tiny machine learning might represent a suitable solution, as it would allow to improve the scalability of models without compromising their overall accuracy. At the same time, the *empirical software engineering* research community might play a key role by understanding the practitioners' needs and practices, other than tailoring (novel) automated approaches on them.
- **Challenge S.7 - Improving the Generalizability of Existing Automated Approaches.** The last challenge in this category concerns with the generalizability of existing approaches that support sustainability analysis of machine learning-enabled systems. Multiple primary studies identified limitations and challenges that the research community should further consider. Similarly to **R.8**, current approaches can only provide support for the development of a limited set of deep neural networks [S17, S21], are built using limited datasets [S34], or are not integrated within MLOps pipelines [S35]. In other terms, the challenge emphasizes the need for novel, more accurate, generalizable, and integrated instruments that may support practitioners throughout the software lifecycle.

Since most of the challenges described pertained to social sustainability, these were investigated in the *'Decision Making'* and *'Healthcare'* domains, which are the most demanding in terms of ethics and fairness. However, we observed a growing interest in the *'Cloud'*, *'IoT Devices'*, and *'Edge Devices'* domains, mostly due to the increasing efforts on energy consumption matters.

Usability (U): Usability is the non-functional requirement for which we could identify fewer challenges. Only a few works targeted this matter, hence suggesting that further research might be worthwhile. Indeed, usability, interpretability, and visualization of both shallow and deep learning models pose significant challenges when it turns to assess quality assurance throughout the software lifecycle. For instance, the lack of interpretability increases the effort required to estimate the soundness of machine learning-enabled systems. Current literature somehow 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 [S14, S37–S39, S59, S68]. Interestingly, the available primary studies did not focus on any specific application domain and proposed general-purpose interpretability models. This further suggests the investigation of domain-specific approaches and methods.

Cross-cutting challenges (C): Besides the challenges pertaining to the individual classes of non-functional requirements, we also identified a set of cross-cutting challenges which are related to the management and assessment of non-functional requirements.

- **Challenge C.1 - Context-based Trade-off Identification.** Finding a trade-off between multiple non-functional requirements represents a key socio-technical and managerial challenge [S8, S13, S16, S16, S17]. On the one hand, machine learning-enabled systems should always preserve accuracy. On the other hand, other non-functional requirements may play a key role depending on the specific context where the system should be employed. As an example, robustness and security might be of paramount importance in the context of driving systems, while fairness might be preferred when dealing with healthcare systems. The identification of the non-functional requirements to preserve may indeed depend on contextual analyses performed at both socio-technical and managerial levels. In the former, novel approaches able to (semi-)automatically mine contextual information that provide practitioners with insights into the non-functional requirements to consider in the specific context might be worthwhile. In the latter, novel software project management methods to assess the significance of non-functional requirements might support the managerial decisions on how to design a machine learning-enabled system. This challenge therefore emphasizes the need for novel methods and instruments to identify the non-functional requirements to simultaneously consider within a given context.
- **Challenge C.2 - Prioritizing and Balancing Non-Functional Requirements.** A second cross-cutting challenge pertains to the balancing of multiple non-functional requirements. While the trade-off identification may provide practitioners with information on the aspects to take into account while developing a ML-enabled system, this would not be enough to prioritize and balance them. Literature argues the need for novel methods to support the technical management of non-functional requirements in terms of prioritization and balancing of non-functional requirements during the optimization process [S1, S13–S15, S18, S50]. As such, the *requirements engineering* research community might further explore this matter, by proposing guidelines or automated instruments to (1) prioritize non-functional requirements and (2) find an optimal balance among both correlated and contrasting objectives by means of the analysis of contextual factors and practitioners' preferences.
- **Challenge C.3 - Software Analytics for Non-Functional Requirement Assessment.** As a last challenge of this category, multiple primary studies advocated the need to empower requirements engineering processes with software analytics instruments able to assess the implications that trade-off choices may have on the the development of ML-enabled systems

[S51, S52, S54, S68, S69]. More particularly, current literature argues the need for software metrics able to inform practitioners of how their requirements engineering decisions may influence both the complexity of the development process and the overall quality of the system being developed. As an example, the definition of strategies to recommend the quality assurance mechanisms to put in place based on the trade-off to meet would be desirable [S69]. On a similar note, researchers claimed that the definition of novel predictive analytics instruments might largely improve requirements engineering processes by enhancing the practitioner's capabilities to assess the impact that trade-off analysis may have on the ML-enabled system throughout the software lifecycle. As a consequence, this challenge may be of the interest of *requirements engineering* and *software analytics* research communities, which may jointly collaborate toward an improved understanding of the support required by requirements engineers in practice.

☞ **Answer to $\mathbf{RQ}_2$.** As a result of $\mathbf{RQ}_2$, we elicited more than 20 software engineering challenges targeting both the individual categories of non-functional requirements identified in our systematic synthesis work and the cross-cutting aspects affecting non-functional requirements engineering as a whole. We methodically untangled the challenges plaguing machine learning-enabled systems, providing insights into the specific research fields interested in those challenges. According to our results, we call for comprehensive analyses and approaches that may assess the impact and implications of the management of individual and multiple non-functional requirements.

4.4 $\mathbf{RQ}_3$ - What automated approaches are proposed to deal with non-functional requirements of ML-enabled systems?

With our last research question ($\mathbf{RQ}_3$), we were interested in surveying the automated approaches to handle non-functional requirements of machine learning-enabled systems proposed so far by researchers. On the one hand, this analysis allowed us to collect information on multiple aspects of those approaches, hence providing insights into the current support available and the related limitations. On the other hand, we could also measure the alignment between the classes of non-functional requirements identified and the approaches available in an effort to understand the extent to which the current approaches fulfil the potential practitioner's needs and, at the same time, what are the limitations of the state of the art and the non-functional requirements that researchers should be more closely pay attention to in future research.

To address our goal, we analyzed the primary studies from three main perspectives. As further discussed in the remainder of the section, all the approaches retrieved were based on machine learning. As such, we first explored how non-functional requirements were measured by the proposed approaches. Second, we analyzed how these machine learning solutions were fed in terms of training datasets and architectural decisions. In the third place, we further deepen our investigation and survey, for each approach, the non-functional requirements addressed and the development phase where it may be applied. By means of these three perspectives, we could basically investigate the design decisions made by researchers when devising machine learning solutions able to address non-functional requirements of machine learning-enabled systems.

4.4.1 On the measurement adopted to analyze non-functional requirements. As a first point of discussion, we analyzed how non-functional requirements were measured by the automated approaches proposed in the literature. Table 6 summarizes the measurement approaches adopted for each non-functional requirement class identified in $\mathbf{RQ}_1$.

We could first observe that existing approaches primarily used software measurements. More particularly, the non-functional requirements pertaining to *'Accuracy'*, *'Efficiency'*, *'Resiliency'*,

Table 6. Non-Functional Requirement Measurement employed by the automated approaches in ML-Enabled System.

Non-Functional Requirement	Measurements approaches
Accuracy	Test set
Efficiency	Nvidia's PyNVML library to measure latency
Maintainability	Software analytics
Resiliency	Perturbed test set, EvalDNN [85] to measure the neuron coverage
Sustainability	Intel Power Gadget, Intel RAPL to measure CPU and GPU energy and CO2. The test set for bias.
Usability	Input sample

'Maintainability', and *'Usability'* were typically assessed by means of software measurements performed on data. On the contrary, the non-functional requirements pertaining to *'Sustainability'* were also assessed using hardware components or libraries that enable the computation of hardware-specific properties. In this sense, we may conclude that addressing non-functional requirements of ML-enabled systems may require a mixture of measurement approaches: this is especially relevant for researchers interested in devising multi-objective optimization techniques, who might indeed need to acquire specific hardware components or combine software and hardware measurements to enable a proper measurement of the attributes to optimize.

Going deeper on the computation of non-functional requirements, *accuracy* was assessed by executing a machine learning model against a test set: this enabled the analysis of (1) true positives and negatives, and (2) false positives and negatives, which were combined to compute the overall accuracy of the model. A similar discussion could be made for the non-functional requirements of the *'Resiliency'* class: to assess properties like security, reliability, and robustness, the approaches proposed in literature proposed to inject perturbations within the dataset, e.g., altered pixels or rotated images for datasets involving images, to assess the model's responsiveness to these diverse inputs. Alternatively, certain approaches assessed the internal behaviour of the model using the EvalDNN framework [85] to evaluate neuron coverage.

Among the non-functional requirements of the *'Usability'*, we could observe that only *explainability* was actually addressed by the currently available automated approaches. To assess *explainability*, these approaches run the machine learning model against a test set, generating input samples that measured the impact of each feature on the resulting performance. As for the non-functional requirements within the *'Maintainability'* class, the approaches adopted software analytics measurements, e.g., by analyzing the training data to identify potential maintainability concerns.

Within *'Sustainability'*, researchers have been employing both software and hardware measurements, depending on the specific non-functional requirement they had to assess. In particular, *fairness* was measured by means of the analysis of the predictions made by the machine learning model, i.e., through a software-based approach: the model was first assessed against a test set and, afterwards, software analytics instruments were adopted to estimate the bias of the predictions made. As for *energy consumption*, the automated approaches adopted methodologies grounded in hardware energy measurement tools, e.g., through the so-called *Running Average Power Limit* (RAPL), which allows monitoring energy consumption of the CPU chip, attached DRAM and on-chip GPU. With these tools, researchers quantify the carbon emissions through mathematical derivations from analysing the actual emissions released by the machine learning model. Finally, when considering the *'Efficiency'* class, the available approaches employed software tools to measure the response latency of forecasts and estimate the response time to get a prediction to output.

Table 7. Overview of the datasets employed by the automated approaches proposed in the literature.

Domain	Non-Functional Requirement	Dataset
Computer Vision	Efficiency	CIFAR-10 - CIFAR-100 [44], MNIST [48], UCF101 [82], ImageNet [23, 45], Dataset X, GTSRB [83], SVHN [68]
	Maintainability	CelebA [52], UnityEyes [98], CIFAR-10 - CIFAR-100 [44], MNIST [48], MIT Indoor Scenes[72], Stanford 40 Actions [102], Stanford Dogs [21]
	Resiliency	CIFAR-10 - CIFAR-100 [44], MNIST [48], CelebA [52], UnityEyes [98], UCF101 [82], SVHN [68], GTSRB [83], Youtube Faces [97], ImageNet [23, 45], Fashion MNIST[100], SVHN [68], Credit Scoring System, Driving [86], MIT Indoor Scenes[72],Stanford 40 Actions [102], Stanford Dogs [21]
	Sustainability	CIFAR-10 - CIFAR-100 [44], MNIST [48], Fashion MNIST[100]
	Usability	CIFAR-10 - CIFAR-100 [44], MNIST [48], , UCF101 [82], GTSRB [83], Youtube Faces [97], Credit Scoring System
Decision Making	Efficiency	Adult [2], Bank Marketing [65], Statlog [1], Compas,[1] Mep,[5] CIFAR-10 - CIFAR-100 [44], MNIST [48]
	Maintainability	Adult [2],Bank Marketing [65]
	Resiliency	Adult [2], Bank Marketing [65], Statlog [1], Compas,[1] Mep,[5] Heart Disease [3], Breast Cancer Wisconsin (Diagnostic) [58], Depresjon [28], Psykose [39], Violent Crime
	Sustainability	Adult [2], Bank Marketing [65], Statlog [1], Compas,[1] Mep,[5] CIFAR-10 - CIFAR-100 [44], MNIST [48], Heart Disease [3]
	Usability	Adult [2], Bank Marketing [65],Statlog [1], CIFAR-10 - CIFAR-100 [44], MNIST [48], Heart Disease [3]
Natural Language Processing	Efficiency	MoleculeNet [99], MultiUN dataset [26], AgNews [106]
	Maintainability	IMDB reviews [56]
	Resiliency	MoleculeNet [99], Snips [20], AgNews [106], MultiUN dataset [26], IMDB reviews [56], AgNews [106]
	Sustainability	MoleculeNet [99], Snips [20], AgNews [106], MultiUN dataset [26]
Automatic Speech Recognition	Efficiency	AN4 [4], TIMIT [87], TED-LIUM 2 [77], TED-LIUM 3 [35]
	Resiliency	AN4 [4], TIMIT [87], TED-LIUM 2 [77], TED-LIUM 3 [35]
Anomaly Detection	Maintainability	SICS Geek Lounge[6]
	Resiliency	SICS Geek Lounge[6]

[1] Compas: https://github.com/propublica/compas-analysis
[2] Titanic: https://www.kaggle.com/c/titanic/data
[3] Bank Execution: https://data.world/markmarkoh/executions-since-1977
[4] Fraud Detection: https://www.kaggle.com/c/frauddetection/data&MarketingCampaigns
[5] Mep: https://meps.ahrq.gov/mepsweb/data_stats/download_data_files_detail.jsp?cboPufNumber=HC-181
[6] 4SICS Geek Lounge: https://www.netresec.com/?page=PCAP4SICS
[7] BTS prediction: https://github.com/rdsea/IoTCloudSamples/tree/master/MLUnits/

4.4.2 On the training decisions behind automated approaches. Tables 7 and 8 report the results achieved when considering how the automated approaches considered were fed. More particularly, Table 7 overviews the datasets retrieved in our study. The datasets are first grouped according to the domain in which the primary study focused. The domains are those resulting from $\mathbf{RQ}_1$. The datasets are then further grouped by class of non-functional requirements they can be experiment with. Finally, the last column of Table 7 provides names and references of the datasets—the reader may observe the missing reference to the *'Preductive Healthcare'*, *'Violent Crime'*,*'Dataset X'*, *'Credit Scoring System'*, and *'AN4'* datasets: they are currently not accessible or have restricted access. At the same time, Table 8 maps the model architectures employed by the automated approaches, namely the machine learning algorithms exploited when building them.

Table 8. Overview of Machine Learning Models employed by the automated approaches proposed in the literature.

Domain	Non-Functional Requirement	Model Architecture
Computer Vision	Efficiency	LSTM, VGG16+LSTM, AlexNet, ResNet18, ResNet-20, ResNet-32, ResNet-56, MobileNet, ShuffleNet, ResNet, VGG16
	Maintanability	ResNet18, ResNet-20, ResNet-32, ResNet-56, MobileNet, MobileNetV2, DenseNet, LSTM, VGG16+LSTM, AlexNet
	Resiliency	AllConvNet, WideResNet, DenseNet, LSTM, ResNet18, ResNet-20, ResNet-32, ResNet50, MobileNet, MobileNetV2, VGG16+LSTM, AlexNet, VGG11, VGG16, ResNet18, GoogleNet, VGG19, LSTM+BiLSTM, LSTM+GRU, MobileNet, ShuffleNet, ResNet-56, DAVE-2, Decision Tree, Random Forest, XGBoost, Udacity, Autumn, Chauffeur, Rwightman
	Sustainability	LSTM+BiLSTM, LSTM+GRU
	Usability	LSTM, VGG16+LSTM, VGG11, VGG16, ResNet18, GoogleNet, Decision Tree, Random Forest, XGBoost
Decision Making	Efficiency	ResNet-18, Naive Model, Sample Clustering, Logistic Regression, Support Vector Machine, Random Forest
	Resiliency	Distributed: Privacy-Preserving Distributed Extremely Randomized Trees, ID3, Centralized: ERT, Random Forest, XGBoost, Decision Tree, Linear SVM, Logistic Regression, Support Vector Machine, Fair SVM, Ensemble Voting Classifier
	Sustainability	ResNet-18, Logistic Regression, Random Forest, Support Vector Machine, Decision Tree, Discriminant Analysis, Naive Model, Sample Clustering, Fair SVM, Ensemble Voting Classifier, KMeans clustering, Agglomerative clustering, Spectral clustering
	Usability	ResNet-18, Logistic Regression, Random Forest, Support Vector Machine, Decision Tree, Discriminant Analysis, Naive Model, Sample Clustering
Natural Language Processing	Efficiency	LSTM, H-NLP, AllenAI, T5, BERT, RoBERTa
	Maintanability	Bert, RoBERTa
	Resiliency	LSTM, BiLSTM+GRU, LSTM+BiLSTM, H-NLP, AllenAI, T5, Bert, RoBERTa
	Sustainability	BiLSTM+GRU, LSTM+BiLSTM, H-NLP, AllenAI, T5
	Usability	LSTM
Automatic Speech Recognition	Efficiency	LSTM, RNN-RNN, Transformer-AED, Conformer-AED
	Resiliency	Bi-LSTM, RNN-RNN, LSTM, Transformer-AED, Conformer-AED

From Table 7, we could first observe a mismatch between the application domains identified in **RQ**$_1$ and those identified here. In the first place, this finding can already provide indications of the comprehensiveness of the automated approaches proposed so far: these only exploited datasets pertaining to a few domains such as *'Computer Vision'*, *'Decision Making'*, *'Natural Language Processing'*, and *'Automated Speech Recognition'*, hence possibly neglecting the additional domains identified in our review. At the same time, our finding may be a reflection of a lack of datasets that can be used to comprehensively investigate non-functional requirements of machine learning-enabled systems—our work focused on the datasets exploited by automated approaches rather than the full set of datasets available in the literature; hence we cannot provide conclusive remarks on the unavailability of datasets; as such, we limit ourselves to highlight a potential limitation that should be further investigated.

Looking at the second column of Table 7, we could observe that multiple datasets were considered for the analysis of the various classes of non-functional requirements, with *efficiency* and *resiliency* having the highest amount of related datasets. From this perspective, we could conclude that multiple datasets exist for experimenting with non-functional requirements of ML-enabled systems and might be further explored in the context of further research on the matter. Yet, we still point out that the automated approaches proposed in the literature missed the analysis of a large number of application domains, which might therefore require further attention in the future.

When considering Table 8, we could first confirm that some specific domains deserved more attention from the research community. Application domains such as *'Computer Vision'* and *'Decision Making'* were indeed confirmed to be the most investigated ones. Furthermore, researchers in these fields have experimented with multiple alternative solutions, assessing the suitability of several algorithms to deal with *efficiency* and *resiliency*. Interestingly, as shown in Table 8, the *'Computer Vision'* domain has been investigated through various solutions based on both convolutional and recurrent neural networks, while the other domains did not have broader investigations. The pieces of information provided by our study could inform researchers on the current state of the artificial intelligence algorithms applied to deal with non-functional requirements of machine learning-enabled systems, possibly informing them of additional perspectives to take into account.

By combining the results coming from our analyses, we could also provide some insights into the nature of the datasets available and the capabilities of the current automated approaches. As we learned from Table 8, most of the developed solutions were based on deep learning algorithms fed through the large datasets shown in Table 7. On the one hand, this information highlights that the datasets available were designed to be large enough to train both shallow and deep learning solutions able to deal with non-functional issues of machine learning-enabled systems: in this respect, our findings report that, despite the intrinsic limitations of the available datasets in terms of domains considered, these might be useful to experiment with multiple artificial intelligence solutions, hence representing potentially relevant benchmarks for the research community. On the other hand, deep learning solutions seem to be the most widely employed in most application domains, possibly indicating that the management of non-functional requirements of machine learning-enabled systems requires advanced learning techniques to properly support practitioners: along this side, further experimentation and comparisons between traditional and deep learning techniques might be worth to be conducted.

4.4.3 On the automated approaches and their relevance. Table 9 lists the specific approaches proposed so far to deal with non-functional requirements of machine learning-enabled systems. The table classifies each approach based on multiple characteristics such as (1) the development phase it can be applied: we classified them based on whether they can be applied in the pre-processing stage of *'Data Engineering'* (Column 'DE' in Table 9), in the in-processing stage of *'Training'* ('T' in Table 9), and in the post-processing stage of *'Post-training'* ('PT' in Table 9); (2) the non-functional requirements targeted, i.e., a '✅' symbol indicates that an approach considered the non-functional requirement, while a '✖' symbol indicates that the approach had not taken a non-functional requirements into account. The table also reports a description of the currently available approaches, hence providing the reader with a quick summary of the capabilities of each of them. Looking at the table, we could first highlight that most of the efforts devoted by the SE4AI community focused on post-processing approaches, with a particular focus on software testing and automated program repair. Pre- and in-processing approaches are present in a lower extent, possibly suggesting that future research might target those aspects to further support practitioners. Secondly, we could also find out that accuracy is the most widely-considered non-functional requirement: as a matter of fact, the vast majority of the automated approaches focused on accuracy as a primary objective to safeguard—this indicates that accuracy is still the most relevant non-functional property to consider while developing automated approaches. Nonetheless, we could also observe a growing interest into the multi-objective optimization of non-functional requirements. In this respect, all the automated approaches available in literature considered non-functional requirements belonging to different classes as *conflicting* objectives and attempted to find a compromise among them.

One of the most frequent multi-objective optimization concerned *accuracy* and *fairness*: researchers have been discovering that an increase in terms of fairness typically leads to a decrease of accuracy [S34, S42]. As a consequence, the approaches proposed attempted to devise solutions that may work on data to let the resulting model be accurate enough while preserving fairness. An example of these multi-objective optimization approaches is the Fair-SMOTE algorithm proposed by Chakraborty et al. [S42], which removes biased labels and rebalances internal distributions so that, based on sensitive attribute, examples are equal in positive and negative classes.

Similar contrasting objectives have been discussed with respect to the compromise between *accuracy* and *resiliency* [S2, S8–S12, S20, S36], *accuracy* and *maintainability* [S16, S23, S25, S62]. Also in these cases, researchers found that the accuracy of ML-enabled systems typically decreases when other non-functional attributes are considered. These observations justify and stimulate additional research on multi-objective optimization.

According to our analysis of the primary studies, we could also find out that non-functional requirements pertaining to the same class were often addressed collectively, showcasing how improvements in one non-functional requirement may have indirectly improved others. For instance, automated approaches aiming at mitigating bias to enhance *fairness* could indirectly improve *accountability* and *ethics* [S34, S66]. Similarly, efforts to boost *robustness* against adversarial samples could indirectly benefit *security*, *flexibility*, *reliability*, and *availability* [S2, S48]. In other terms, some of the currently available approaches may have *direct* and *indirect* implications on non-functional requirements. However, we could not find a general rule describing the indirect effect of the optimization of each non-functional requirement: we believe that further software analytics research might explore this matter, possibly providing a taxonomy able to explain the cause-effect relations among non-functional requirements.

Still in terms of indirect optimization, it is worth discussing the case of *capacity*, i.e., the amount of space required to store the model and any associated data. The optimization of this non-functional requirement leads to a model reduction where an algorithm reduces the overall size of the model, producing a smaller model that is later distributed. By analyzing the primary studies proposing model reduction techniques [S1, S37, S50], we could observe that *capacity* has not only indirect effects on non-functional requirements pertaining to its class (*'Efficiency'*), but also to other classes such as *'Sustainability'* and *'Accuracy'*. Indeed, the optimization of *capacity* typically leads to improvements in terms of *performance*, i.e., a smaller model has a more efficient throughput bound. At the same time, this has implications for the overall *energy consumption* of the model, i.e., a smaller model consumes less energy. Yet, optimizing *capacity* may imply a reduction of *accuracy.*

The case of *capacity* further justifies the need for additional research on the relations among non-functional requirements and, as such, ours represents a call for *requirements engineering*, *software analytics*, and *empirical software engineering* researchers.

As an additional perspective, we could understand that there is no approach able to comprehensively analyze the whole set of non-functional requirements classified in the context of $\mathbf{RQ}_1$. As a matter of fact, only a few approaches optimized two or more non-functional requirements and, more interestingly, only a few approaches were deemed to be relevant for non-functional aspects such as efficiency, maintainability, and usability, hence suggesting that more research should be done on the optimization of those concerns. Furthermore, we could see that many of the challenges identified in $\mathbf{RQ}_2$ are still not addressed by currently available approaches. In other terms, while the literature called for research in various non-functional requirements domains, there still seems to be a little automated tooling able to address it. For instance, we could observe little to no automated solutions available for solving privacy, post-deployment simulations, legal aspects, and adversarial learning. As such, the final outcome of our literature review points out, in the first place, the lack of automated approaches able to comprehensively consider the non-functional requirements of machine learning-enabled systems. Also, we let emerge the need for additional studies on the relations among non-functional requirements, which may effectively drive the definition of multi-objective optimization approaches. Finally, our work highlights that most of the challenges identified in the context of $\mathbf{RQ}_2$ are still open and serve as a way to inform our research community on the next steps to perform to provide additional support to practitioners.

Table 9. Automated approach to optimize non-functional requirements of ML-enabled systems. ‘DE’=Data Engineering; ‘T’=Training; ‘PT’=Post-Training; ‘A’=Accuracy; ‘E’=Efficiency; ‘M’=Maintainability; ‘R’=Resiliency; ‘S’=Sustainability; ‘U’=Usability.

DE	T	PT	Approach	Description	A	E	M	R	S	U
✓	✓	✗	Fairway [S34]	This paper discusses the issue of algorithmic discrimination and ethical bias in machine learning software. It proposes a method called Fairway to detect and mitigate bias in the training data and the learned model. The method combines pre-processing and in-processing approaches and removes ethical bias without significantly affecting predictive accuracy. The paper advocates for testing and mitigating bias to become a routine part of the machine learning software development life cycle	✓	✗	✗	✗	✓	✓
✓	✓	✗	Deepstate [S20]	The paper proposes DeepState, a test suite selection tool for RNN models to reduce data labeling and computation costs. DeepState selects data based on a stateful perspective of RNN and identifies possibly misclassified tests by capturing neuron state changes. The tool further provides a test selection method to obtain a test suite with strong fault detection and model improvement capabilities	✓	✗	✗	✓	✓	✗
✓	✓	✗	Dare [S12]	The paper proposes a novel model training framework called Dare that aims to improve the universal robustness of DL models. The proposed approach incorporates model transformation and data augmentation in a delta debugging fashion	✓	✗	✗	✓	✗	✗
✓	✓	✗	MAAT [S13]	This paper proposes a novel ensemble approach, MAAT, to improve the fairness-accuracy trade-off for ML software. MAAT combines models optimized for different objectives: fairness and ML accuracy.	✓	✗	✗	✗	✓	✗
✗	✓	✗	Parfait-ML [S21]	This paper investigates the impact of hyperparameters on fairness in ML algorithms. The authors propose three search-based software testing algorithms to identify hyperparameters that can improve fairness without sacrificing precision. They use statistical debugging to explain the role of these parameters in improving fairness and implement the proposed approach in the tool Parfait-ML.	✓	✗	✗	✗	✓	✓
✗	✓	✗	Decomposing Convolutional Neural Networks into Reusable and Replaceable Modules [S25]	The paper proposes a method to decompose a CNN model for image classification into modules for each output class, enabling reusability and replaceability. Reusing and replacing these modules reduces CO2e emissions	✓	✓	✓	✗	✓	✗
✗	✓	✗	k-PPD-ERT [S36]	This paper proposes a distributed extremely randomized trees (ERT) algorithm for machine learning on healthcare data without compromising patients’ privacy. The proposed method, k-PPD-ERT, uses a cloud platform and is tested on medical datasets, including Heart Disease, Breast Cancer, and two mental health datasets.	✓	✗	✗	✓	✓	✗
✗	✓	✗	DeepCorrect [S10]	This paper discusses the impact of image distortions, such as blur and noise, on pre-trained convolutional filters in deep neural networks used for computer vision tasks. They introduced a new approach called DeepCorrect. It applies small stacks of convolutional layers with residual connections to correct the worst affected filter activations without changing the rest of the pre-trained filter outputs. The accuracy results demonstrate that DeepCorrect significantly improves the robustness of DNNs against distorted images	✓	✗	✗	✓	✗	✗
✗	✓	✗	Distributed Selective SGD [S8]	The paper proposes a system that allows multiple parties to learn an accurate neural network model without sharing their datasets by selectively sharing small subsets of their models’ key parameters during training. This system preserves privacy while still improving learning accuracy beyond what is achievable on their inputs	✓	✗	✗	✓	✗	✗

Continued on next page

Table 9 – continued from previous page

DE	T	PT	Approach	Description	A	E	M	R	S	U
✓	✓	✗	Towards Training Reproducible Deep Learning Models [S16]	The paper proposes a systematic approach to training reproducible DL models, which is crucial for various tasks like training, testing, debugging, and auditing. DL models are challenging to reproduce due to issues like randomness in the software and non-determinism in the hardware. The proposed approach includes general criteria to evaluate reproducibility, a framework using record-and-replay and profile-and-patch techniques to mitigate software-related randomness and hardware-related non-determinism, and a reproducibility guideline	✓	✗	✓	✗	✗	✗
✗	✓	✓	NNREPAIR [S2]	NNREPAIR tool use a technique for repairing neural network classifiers by fixing potentially faulty network parameters. It uses fault localization to identify the faulty parameters and then performs repairs using constraint solving to make small parameter modifications. The technique can improve the accuracy of models, fix security vulnerabilities caused by the poisoning of training data, and improve the robustness of the network against adversarial attacks.	✓	✗	✗	✓	✗	✗
✗	✓	✓	AI-Lancet [S9]	AI-Lancet proposes a systematic approach to trace and fixes errors in deep learning models that could lead to severe security issues. They locate error-inducing neurons and propose two methods, neuron-flip and neuron-fine-tuning, to fix the errors.	✓	✗	✗	✓	✗	✓
✗	✓	✓	Apricot [S11]	The paper presents Apricot, a novel weight-adaptation approach to fixing imprecise deep learning models. Apricot generates a set of reduced DL models from the original DL model. In each iteration, it adjusts the weights of the input DL model towards the average weight of the reduced DL models that correctly classify a failing test case and/or away from that of the reduced DL models that misclassify the same test case.	✓	✗	✗	✓	✗	✗
✗	✓	✓	FairNeuron [S17]	The paper proposes a tool called FairNeuron, which automatically repairs DNN models to balance accuracy and fairness trade-offs by detecting neurons with contradictory optimization directions from accuracy and fairness training goals and achieving a trade-off by selective dropout. The approach is lightweight, scales to large models, and is more efficient than state-of-the-art methods.	✓	✗	✗	✓	✓	✓
✗	✗	✓	RobOT [S23]	This paper proposes a testing framework called Robustness-Oriented Testing (RobOT) for DL systems that aims to improve model robustness. RobOT utilizes a novel quantitative measurement that evaluates the value of each test case in improving model robustness and the convergence quality of the model robustness improvement.	✓	✗	✓	✓	✗	✗
✗	✓	✓	Search-based test and improvement of machine-learning-based anomaly detection systems [S62]	This paper focuses on testing intrusion detection systems (IDS) against training attacks, which are designed to exploit the vulnerability of machine-learning-based anomaly detection systems. The authors propose a search-based approach to test IDS by making training attacks, and they also propose searching for countermeasures to increase the resilience of the tested IDS.	✗	✗	✓	✓	✗	✗
✗	✗	✓	ASRTest [S28]	The paper proposes a testing approach for deep neural network-driven automatic speech recognition (ASR) systems called ASRTest, which is built on the theory of metamorphic testing. The approach involves designing metamorphic relations for ASR systems and implementing transformation operators to simulate practical scenarios that generate speeches. Gini impurity is used to guide the generation process and improve testing efficiency.	✓	✓	✗	✓	✗	✗
✗	✓	✓	Hybrid Repair [S29]	The paper proposes HybridRepair, an annotation-efficient solution for repairing deep learning models with a mismatch between training data and field data distribution. The solution takes a holistic approach that combines a small amount of annotated data with a large amount of unlabeled data, leveraging semi-supervised learning techniques to boost training data density. The proposed method selectively annotates data in failure regions and propagates labels to neighboring data while using a selected part of the unlabeled data to avoid the impact of distribution shift on SSL solutions.	✓	✗	✗	✓	✗	✗
✗	✗	✓	BET [S30]	The paper proposes a Black-box Efficient Testing (BET) method for identifying defects in CNNs without requiring full knowledge of the target model. The BET method generates continuous perturbations in a black-box manner and uses a tunable objective function to guide the testing process. The method also employs an efficiency-centric policy to find error-inducing inputs within a fixed query budget	✓	✗	✗	✓	✗	✗

Continued on next page

Table 9 – continued from previous page

DE	T	PT	Approach	Description	A	E	M	R	S	U
✖	✔	✔	Security for distributed deep neural networks: Towards data confidentiality & intellectual property protection [S32]	This paper proposes an approach for protecting distributed DNN based software assets, including the confidentiality of input and output data streams and safeguarding intellectual property, using fully homomorphic encryption (FHE). The authors evaluate the feasibility of this approach on a convolutional neural network (CNN) for image classification deployed.	✔	✔	✖	✔	✖	✖
✖	✖	✔	Interpreting Deep Learning-based Vulnerability Detector Predictions Based on Heuristic Searching [S38]	The article proposes a new explanation framework to identify the small number of tokens that contribute significantly to the detector's prediction, which can help domain experts understand and accept the detector's outputs. The experiments show that this framework has higher fidelity than existing methods, especially in real-world scenarios where features are not independent.	✖	✖	✖	✔	✖	✔
✖	✖	✔	LEMNA [S39]	The paper proposes a high-fidelity explanation method called LEMNA for security applications, which generates a small set of interpretable features to explain how the input sample is classified. LEMNA approximates a local area of the deep learning decision boundary using a simple interpretable model that handles feature dependency and nonlinear local boundaries to improve explanation fidelity.	✖	✖	✖	✔	✔	✔
✖	✔	✔	HUDD [S3]	Heatmap-based unsupervised debugging of DNNs (HUDD) uses heatmaps and clustering to identify root causes of DNN errors automatically and supports retraining with related images. Evaluation in the automotive domain showed HUDD effectively identified root causes and improved DNN accuracy.	✔	✖	✖	✔	✖	✖
✖	✔	✔	DeepRepair [S4]	To resolve errors in deep neural networks caused by a mismatch with real-world conditions after deployment, the proposed solution is style-driven data augmentation. DeepRepair, using style transfer, incorporates unknown error patterns into the training data. It successfully repairs convolutional and recurrent neural networks while maintaining better accuracy on clean datasets.	✔	✖	✖	✔	✖	✖
✖	✖	✔	PLUM [S5]	To address inaccuracies in DL models, Plum is introduced as a novel hyperheuristic approach. Plum generates a set of DL model candidates using low-level repair strategies, evaluates them based on their fixing effects, and prioritizes the best strategy to output a fixed DL model.	✔	✖	✔	✔	✖	✖
✖	✔	✔	TestRNN [S59]	TestRNN is a coverage-guided testing approach for recurrent neural networks (RNNs), specifically focusing on long short-term memory networks (LSTMs). The method employs three test metrics and a genetic algorithm to detect defects by exploring the internal behavior of LSTMs. testRNN demonstrates superior performance in capturing temporal semantics and handling input perturbation, providing meaningful and interpretable testing results..	✔	✖	✔	✔	✖	✔
✖	✖	✔	NMTSloth [S40]	NMTSloth is a technique that exploits a fundamental property in Neural Machine Translation (NMT) systems—computation efficiency is determined by output length. By making subtle changes to input sentences, NMTSloth significantly delays the generation of end-of-sentence tokens, leading to increased response latency and energy consumption in NMT systems. The technique shows substantial impacts on real-world mobile device battery power.	✖	✔	✖	✔	✔	✖
✖	✔	✖	RULER [S41]	RULER is a new model repair technique, by discriminating sensitive and non-sensitive attributes during test case generation for model repair. The generated cases are then used in training to improve DNN fairness. RULER balances the trade-off between accuracy and fairness by decomposing the training procedure into two phases and introducing a novel iterative adversarial training method for fairness	✔	✖	✖	✖	✔	✖
✔	✔	✔	Fair-SMOTE [S42]	The Fair-SMOTE algorithm is introduced to remove biased labels and balance internal data distributions based on sensitive attributes, effectively reducing bias without sacrificing accuracy.	✔	✖	✖	✖	✔	✖
✖	✖	✔	DeepJanus [S43]	DeepJanus assesses the quality of a deep learning system by examining its behavior frontier, i.e., distinct pairs of similar inputs that trigger different responses. Experimental results show that the frontier inputs provide developers with both quantitative and qualitative insights, helping to assess the quality of the DL system and identify problematic inputs.	✔	✖	✖	✔	✖	✖

Continued on next page

Table 9 – continued from previous page

DE	T	PT	Approach	Description	A	E	M	R	S	U
✖	✔	✖	CoEvA2 [S45]	CoEvA2 is a search-based method that generates valid opponent examples that satisfy domain constraints. CoEvA2 uses multi-objective search to simultaneously handle the constraints, execute the attack, and maximize the amount of overdraft required. CoEvA2 generates thousands of valid adversary examples by improving the system based on the examples produced, increasing robustness and making their attack fail.	✔	✖	✖	✔	✖	✖
✔	✔	✖	Reducing DNN labelling cost using surprise adequacy: an industrial case study for autonomous driving [S46]	The paper proposes a technique to reduce manual labeling cost during the development of a DNN-based object segmentation development by using Surprise Adequacy (SA) to predict model performance for unlabeled inputs. Engineers can trade off cost savings against acceptable inaccuracy levels in different development phases and scenarios.	✔	✖	✔	✖	✔	✖
✖	✖	✔	DeepTest [S47]	DeepTest is a systematic testing tool for automatically detecting erroneous behaviors of DNN-driven vehicles that can potentially lead to fatal crashes. First, automatically generated test cases leveraging real-world changes in driving conditions like rain, fog, lighting conditions, etc. DeepTest systematically explores different parts of the DNN logic by generating test inputs that maximize the number of activated neurons. DNN accuracy can be improved by retraining with the synthetic data generated by DeepTest.	✔	✖	✖	✔	✖	✖
✖	✔	✔	Aeqitas [S63]	Aeqitas is an automated approach for validating fairness in machine-learning models used for decision-making in sensitive domains. Aeqitas identifies discriminatory inputs revealing fairness violations using probabilistic search. Leveraging the robustness of common models, it generates scalable tests that, when added to the training set, systematically improve fairness.	✔	✖	✖	✔	✔	✖
✖	✔	✔	Adversarial Discrimination Finder [S64]	Adversarial Discrimination Finder is a gradient-based algorithm for generating individual discriminatory instances in DNN. It has two parts: global generation and local generation. Gradients guide the crafting of discriminatory instances in both stages. The individual discriminatory instances generated by ADF are useful to improve the fairness of the DNN through retraining	✖	✖	✖	✔	✔	✔
✖	✔	✔	DeepRoad [S48]	DeepRoad is an unsupervised framework for testing DNN-based autonomous driving systems. Using generative adversarial networks (GANs), DeepRoad creates scenes with different weather conditions. The framework employs metamorphic testing to assess system consistency using synthetic images and validates input images by measuring their distance from training images using VGGNet features. Experimental results on three DNN-based autonomous driving systems demonstrate DeepRoad's ability to detect many inconsistent behaviors and improve system robustness through effective input validation.	✔	✖	✖	✔	✖	✖
✖	✔	✖	ReMoS [S49]	ReMoS is a relevant model slicing technique to reduce defect inheritance during transfer learning while retaining useful knowledge from the teacher model. Specifically, ReMoS computes a model slice (a subset of model weights) that is relevant to the student task based on the neuron coverage information obtained by profiling the teacher model on the student task. ReMoS can reduce inherited defects effectively and efficiently with minimal sacrifice of accuracy.	✔	✖	✖	✔	✔	✖
✖	✖	✔	SafeCompress [S50]	SafeCompress is a framework for test-driven sparse training. SafeCompress can automatically compress a large model into a small one, following the dynamic sparse training paradigm. Moreover, considering a representative attack, namely the membership inference attack (MIA), they developed a concrete mechanism for safe model compression called MIA-SafeCompress.	✔	✔	✖	✔	✖	✖
✔	✖	✖	Fair Class Balancing [S54]	The paper proposes a fair class balancing method that allows enhanced model fairness without using any information about sensitive attributes. They show that the method can achieve accurate prediction accuracy while concurrently improving fairness.	✔	✖	✖	✖	✔	✖
✔	✔	✔	Clarify [S68]	The paper introduces Amazon SageMaker Clarify, an explainability feature as part of Amazon SageMaker. It addresses the challenge of understanding the predictions and potential biases of machine learning models. Clarify offers insights into data and models by identifying biases and explaining predictions. The feature supports bias detection and computes feature importance throughout the ML lifecycle—during data preparation, model evaluation, and post-deployment monitoring.	✖	✖	✖	✖	✔	✔

☞ **Answer to $\mathbf{RQ}_3$.** Our $\mathbf{RQ}_3$ finally points out that the currently available support is limited and, indeed, most of the software engineering challenges identified are still open and worth further investigation. In addition, the currently available approaches do not often perform a comprehensive optimization of non-functional requirements; as such, practitioners would be required to deploy a complex selection of tools to optimize relevant requirements, not being able to rely on multi-objective approaches that may find trade-offs among desired design aspects. Last but not least, our research let emerge the presence of non-trivial cause-effect relations among non-functional requirements, which should be further explored to drive the definition of novel multi-objective optimization approaches.

5 DISCUSSION AND IMPLICATIONS

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

On the Relation between the acquired knowledge and the previous one. Our results allowed us to discover multiple aspects concerned with non-functional requirements of machine learning-enabled systems that extend the existing knowledge. The outcome of $\mathbf{RQ}_1$ provided a comprehensive classification of non-functional requirements along with their classes. In this respect, our work extended the insights provided by Habibullah et al. [32], providing a larger overview of the issues that practitioners should consider while developing machine learning-enabled systems. Specifically, we noticed three major differences. First, Habibullah et al. [32] did not include *'Sustainability'* as a category of non-functional requirements but, rather, they limited the categorization to aspects connected to ethics and bias. Hence, our work extends the classification by contextualizing the aspects considered by Habibullah et al. [32] within the broader context of sustainability, explicitly referring to three dimensions such as environmental, economic, and social sustainability. Secondly, our systematic analysis led to the identification of four additional non-functional requirements connected to *'Resiliency'*: *robustness*, *reliability*, *behavioral*, and *flexibility*. In this sense, we were able to enlarge the knowledge of the non-functional requirements of ML-enabled systems, building on top of the existing classification schemas to further enrich them with additional characteristics and attributes that should be optimized. Finally, we also noticed a substantial difference in the treatment of the category *'Maintainability'*. Habibullah et al. [32] combined the traditional concept of maintainability, i.e., the ability of a system to be modified, improved, and adapted, with the concept of usability, i.e., how effectively a user can learn and use the system. Based on our systematic work, we found out that the concepts would be better split into two separate categories. The rationale is twofold. On the one hand, maintainability and usability can be logically considered as two different non-functional requirements which pertain to distinct desirable attributes of ML-enabled systems. On the other hand, the optimization of these two attributes would require different methods and strategies and, for this reason, further research would benefit from a clear distinction between the two. The additional knowledge gathered by our work may serve as input to both researchers and practitioners. The former might take our review as a means to design empirical experimentation and approaches able to deal with the non-functional aspects that previous research did not consider. The former might instead take our classification as an input to increase their own awareness with respect to the problems arising when developing machine learning-enabled systems, other than reason on how to address those aspects in real-world use cases.

> **Take Away Message.** Our classification of non-functional requirements may inspire further research on the matter, other than making practitioners aware of the multiple, multi-faceted concerns arising when developing machine learning-enabled software systems.

At the same time, it is also worth commenting on how the findings coming from $\mathbf{RQ}_2$ and $\mathbf{RQ}_3$ inform future research. Our research identified a set of over 20 software engineering challenges that should be further considered by our community. When comparing our findings with those reported by Horkoff [37], we may first observe that the level of granularity of the challenges reported is, in general, different. Indeed, Horkoff [37] discussed a set of *conceptual issues* that may threaten the researchers' capabilities of effectively dealing with non-functional requirements of machine learning-enabled systems. For instance, the author reported that the understanding of the research community with respect to non-functional requirements of machine learning-enabled systems is fragmented and incomplete, which includes the lack of definitions of the key attributes affecting the way machine learning-enabled systems operate. On the one hand, our work was able to address some of the conceptual concerns raised by Horkoff [37], offering a theoretical framework describing the definitions of non-functional requirements, how they have been explored so far, and what are the potential inter-relations among them. On the other hand, the set of challenges identified may be seen as a more concrete instantiation of the conceptual concerns reported by Horkoff [37]. Indeed, we were able to describe the specific challenges that the research community is called to address for each non-functional requirement, hence providing researchers with an actionable instrument to design further studies and investigations into the matter. In addition to that, it is also worth reporting that we could also corroborate one of the challenges identified by Horkoff [37], namely the need for engineering machine learning-enabled systems for reducing internal errors and security. So, in conclusion, we could elaborate a larger, comprehensive catalog of challenges that have to do with multiple aspects of SE4AI and that naturally inform a number of software engineering research communities, from the one on requirements engineering till the one on software testing. As such, our work can be considered as a cross-cutting knowledge base for serving further research in software engineering. The outcomes of $\mathbf{RQ}_3$ reinforce our conclusion: as many non-functional aspects were not still considered or partially taken into account by existing automated approaches, additional research would be worthy. Our overview of both challenges and automated approaches provides a ground for the development of additional software engineering instruments.

> **Take Away Message.** Our work could identify a number of software engineering challenges that are not yet targeted by automated approaches, hence suggesting the future avenues that researchers can consider to further enlarge the knowledge and support provided to handle non-functional requirements of machine learning-enabled systems.

As a last point of discussion, we comment on how we envision our results have influence on the state of the practice. The outcome of $\mathbf{RQ}_1$ may possibly be exploited to support the definition of standards and guidelines that attempt to describe how to deal with non-functional requirements of machine learning-enabled systems. For example, the definitions provided might be potentially useful to support and complement an existing, emerging standard such as the *ISO/IEC 25059*,[14] which defines an initial taxonomy of non-functional properties relevant for machine learning-enabled systems. In this respect, we foresee improved implementations of the standard, which takes our findings into account, hence enlarging the definitions available for researchers and practitioners. Furthermore, when comparing our findings against the research that explored the

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

current state of the industrial practice, we see some interesting complementarities. In particular, Amershi et al. [8] conducted a case study in Microsoft where they let emerge a catalog of best practices to make software engineering actionable for the development of machine learning-enabled systems. Among these practices, the authors pointed out the need for model debugging and interpretability and for model evolution, evaluation, and deployment. In the first place, the outcome of **RQ**$_1$ has the potential to make the best practices identified by Amershi et al. [8] more tangible: indeed, our findings provide a set of factors that strictly relate to the model debugging, interpretability, evolution, evaluation, and deployment which are potentially measurable and, therefore monitorable throughout the different development stages of machine learning-enabled systems. As an example, the *retrainability* attribute pertaining to the *'Maintainability'* cluster may impact the evolution and deployment of machine learning-enabled systems. Our systematic synthesis provides a theoretical foundation to build novel measurement and monitoring systems that may be employed to produce higher-quality machine learning-enabled systems. In addition, the findings of **RQ**$_2$ provide insights into the research gaps making the best practices identified by Amershi et al. [8] difficult to actually implement. Also in this case, our systematic synthesis may therefore provide researchers with an improved understanding of the next steps to pursue to better support the development and evolution of machine learning-enabled systems. Similarly, the automated approaches coming from **RQ**$_3$ may complement the insights by Amershi et al. [8]. On the one hand, our findings offer a comprehensive list of automated approaches that have been proposed in the literature to ease the application of the best practices, along with their potential limitations: the characteristics of these approaches might be assessed by practitioners to understand the limits and potentials of the currently available support, hence evaluating whether these might be actually transferred to industry. On the other hand, our results emphasize the gap between research and practice, indicating that only a few approaches may support practitioners: in this sense, our work may serve as a call to action for researchers.

> **Take Away Message.** Our findings complement the current state of the practice in different manners. First, the outcome of **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 **RQ**$_1$, **RQ**$_2$, and **RQ**$_3$ can be combined to the pieces of information 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. According to our results, and specifically, those coming from **RQ**$_2$, there exist some inter-relations among non-functional requirements. 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 the development of machine learning-enabled systems. In this respect, researchers in the area of requirements engineering and empirical software engineering might cooperate toward the development of novel taxonomies that may map the relations among the non-functional requirements of machine learning-enabled systems and how they impact each other. Our results also called for further research on the identification, management, and assessment of the trade-offs among multiple non-functional requirements. The results of **RQ**$_3$ confirmed the presence of cause-effect relations and indicated that the multi-objective optimization of non-functional requirements represents a growing trend that should be further explored. In this respect, the set of challenges identified in our work could provide insights into the next research avenues to pursue, pinpointing the multi-faceted engineering challenges to face and the non-functional requirements that should be further explored and analyzed by means of requirements engineering, software analytics, and empirical software engineering. At the same time, the results of **RQ**$_3$ further inform researchers and practitioners. On the one hand, we could point out that the available automated approaches could only satisfy subsets of non-functional requirements, hence not being comprehensive enough to properly support the holistic development of machine learning-enabled systems. On the other hand, practitioners could be informed of the support available, possibly highlighting opportunities for technological transfer. In any case, our work represents a call for a multi-objective, comprehensive investigation into the non-functional attributes of machine learning-enabled systems and their interconnections.

> **Take Away Message.** Our findings suggest novel interconnections and research on the multi-objective optimization of non-functional requirements of machine learning-enabled systems. We call for a brand 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 machine learning-enabled systems.

A Managerial Viewpoint. As a follow-up discussion, the research questions targeted by our work raise multiple trade-offs to consider when developing machine learning-enabled systems. As a matter of fact, the accuracy of artificial intelligence algorithms must necessarily be balanced with other aspects to create trustworthy software systems. On the one hand, this clearly represents a call for further research. On the other hand, our work may have implications on the managerial side: the constant and continuous search for trade-offs indeed represents a high-level challenges for project managers, who are required to monitor and handle multiple non-functional attributes throughout the evolution of machine learning-enabled systems. In the first place, the results produced by **RQ**$_1$ and **RQ**$_3$ may allow the reader to understand what is the set of non-functional attributes to actually take into account and the approaches currently available to optimize them, respectively. Secondly, the challenges identified in the context of **RQ**$_2$ should not only be considered from a technical perspective but also from a socio-technical and managerial one. Our work indeed calls for further research on the managerial strategies that would indicate the most appropriate management policies to apply when handling non-functional requirements. Also, we bring to the attention of software engineering and software project management researchers the lack of optimization approaches that may be contextual, dependent, and adaptable to the evolution of the system—we argue that those properties would be key to enabling the proper management of non-functional requirements over the software lifecycle.

> **Take Away Message.** We call for research on managerial and socio-technical strategies to handle non-functional requirements throughout the evolution of machine learning-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.

Machine Learning Sustainability: Software Engineering Comes to Rescue. The results of both **RQ**$_2$ and **RQ**$_3$ identified sustainability as a non-functional requirement that has been somehow neglected by our research community. While previous work has started exploring the social side of sustainability, we could identify a lack of software engineering investigations into energy- and cost-related concerns. On the one hand, our work identified *tiny machine learning* [50] as a promising solution to some of the key challenges affecting the energy consumption of machine learning-enabled systems: as such, we call for additional investigations into these aspects. On the other hand, we also highlight a lack of experimentation in terms of costs: in this respect, further analysis of how to predict and manage costs, other than how to optimize non-functional approaches by taking those aspects into consideration, would be worthwhile.

> **Take Away Message.** Sustainability concerns should be more carefully considered by the software engineering research community. There is a need for approaches to deal with energy consumption other than cost-aware non-functional requirements optimization strategies.

On the Currently Available Automated Support. Both $\mathbf{RQ}_2$ and $\mathbf{RQ}_3$ pointed out the limited amount of automated approaches, other than their low generalizability. We could indeed realize the limited support provided by researchers in terms of the analysis of specific contexts, architectures, and algorithms. We also elaborated on the need for novel approaches that could simultaneously address multiple non-functional requirements: in this sense, the observations provided in the context of $\mathbf{RQ}_3$ might serve as a basis for further research on the matter. At the same time, we elicited additional needs in terms of automated support, especially related to software analytics instruments that could support practitioners while identifying, managing, and assessing the impact of non-functional requirements on both development processes and products. Hence, it is our hope that the findings reported may represent a call for researchers working in the field of automated software engineering to propose novel techniques and solutions able to support practitioners in a broader manner and in a way that faces the specific challenges pertaining to the management of non-functional requirements.

> **Take Away Message.** Our findings clearly point out the limited automated support that practitioners have, other than the limited generalizability of the available solutions. Our work represents a call for software analytics and automated software engineering research efforts aiming at facing the specific challenges identified and providing novel, generalizable approaches to handle and optimize non-functional requirements of machine learning systems in practice.

6 THREATS TO VALIDITY

The validity of both the research method and conclusions drawn in our systematic literature review might have been threatened by some key considerations. This section overviews the potential limitations of our study and how these were mitigated when designing the study.

Literature selection. A critical challenge for a systematic literature review is identifying a consistent and comprehensive body of knowledge concerning the subject of interest. In this respect, we first approached the search process by adhering to well-established guidelines [42]. Nonetheless, we realized that these guidelines would not have been sufficient to extract the required body of knowledge and, as a consequence, might have led to missing key resources. To mitigate this threat, we, therefore, opted for a *hybrid* systematic analysis [66]: we complemented the set of primary studies identified through the guidelines by Kitchenham et al. [42] with additional resources coming from (1) the systematic screening of the research articles published in top-tier software engineering and artificial intelligence venues [96]; and (2) multiple snowballing iterations of the incoming and outcoming references of the primary studies identified [95]. In this respect, it is worth further discussing the seed search procedure conducted on artificial intelligence venues. As explained in Section 3.3, we had to limit ourselves to the analysis of a subset of all the top-tier conferences and journals in the field of artificial intelligence to make systematic scanning sustainable and feasible. In doing so, we solely considered the venues having a higher likelihood to include engineering or empirical pieces of work that might have been relevant for addressing our research questions: among 35,155 papers published within the selected venues between 2012 and 2022, we only found 12 potentially relevant articles, which reduced to seven after the quality assessment stage. We are aware that an extensive analysis of the artificial intelligence venues might have identified some additional relevant primary studies: this remains a limitation of our work that the reader must be aware of. However, our analyses revealed that the amount of papers published in artificial intelligence venues that describe or optimize non-functional requirements is somehow limited (only 0.02% of the papers scanned were finally included), possibly not justifying the effort that would have been required. In this sense, alternative research instruments, e.g., qualitative surveys or interviews with machine learning engineers, might be more useful to complement our findings and identify additional concerns or approaches used to optimize non-functional requirements.

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 [60] 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 [96].

Literature analysis and synthesis. Upon completing the search process, we applied several steps to ensure the inclusion of the relevant primary studies to address the research questions driving our systematic literature review. In the first place, we designed a set of exclusion criteria with the intent of filtering out the primary studies that did not fit our scope. In addition, we proceeded with the definition of inclusion criteria, other than a formal quality assessment of the suitability of each identified resource for the goals of our study.

These steps were conducted manually; therefore, the risk of subjectiveness and human error were the main limitations. While the first author mainly conducted all the activities, there are two considerations to make. In the first place, we designed the systematic literature review so that validation was conducted at the end of each step. The validation first involved the second author of the paper, who took the role of inspector: he was indeed called to verify the actions conducted by the first author and identify potential errors. At the end of this process, both authors opened a discussion and addressed the concerns raised in the validation.

Secondly, the two authors constantly worked together to define the research method and actions to make the study sound and reliable. In particular, they met weekly to discuss the advances of the systematic literature review and the potential limitations to address. These weekly meetings further limited potential concerns due to subjectiveness and human bias. Nonetheless, we acknowledge that the entire process has been conducted based on the knowledge and expertise of the authors, which might be limited. On the one hand, we released all the material produced to be as transparent as possible [60]. On the other hand, we can only report some background information - this may be useful to assess the study's validity and estimate the resources required to replicate our work. The two authors have a research experience of one and ten 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, optimization of non-functional requirements, and software maintenance and evolution. 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) - the second author is the lecturer of those courses, while the first is a teaching assistant.

The data analysis of the primary studies allowed us to address the research questions of the study, providing a picture of the current state of the art, synthesizing the research efforts conducted so far, identifying the areas that are still neglected or unexplored, and pointing out the next research directions that the software engineering research community should pursue to better assist practitioners in their activities. In doing so, we could analyze a bit more than two primary studies for each non-functional requirement, on average (69 articles concerned with 30 non-functional requirements). While this might indicate the limited generalizability of our findings, we do not see this point as a real limitation: our systematic review has indeed an intrinsic value which is independent of the number of primary resources finally considered. We are aware that future research may classify additional non-functional requirements or identify further challenges which are not documented in this paper: our ultimate goal is exactly that of stimulating more research on the matter by providing a foundational basis describing the current body of knowledge and the limitations that should be addressed in the future.

7 CONCLUSION

In this paper, we conducted a hybrid systematic literature review on non-functional requirements of machine learning-enabled systems—this was motivated by the increasing number of papers published on these aspects during the last ten years. We tackled three main research angles such as (1) the classification of the non-functional requirements investigated so far, (2) the challenges to face when dealing with them, and (3) the automated approaches proposed in literature to support practitioners when optimizing them in practice. The results of the systematic literature review do not only summarized the current knowledge on the matter, but also opened new horizons and challenges that our research community should consider. It is our hope that researchers and fresh Ph.D. Students might be inspired by our work and, further contributing to the field. All in all, our work brings the following major contributions:

(1) A systematic literature review on non-functional attributes of machine learning-intensive systems, which allowed to systematically classify them, other than identify the current open challenges and approaches to cope with them. These pieces of information may be exploited by researchers to define the next research steps to improve the currently available techniques and overall support provided to practitioners;

(2) A set of implications and take-away messages that researchers may use to address future research avenues on the management and optimization of non-functional requirements of machine learning-enabled systems;

(3) An online appendix containing all data and scripts used in the study, which might be used to extend the scope of the study or replicate our work.

The considerations and implications of this systematic literature review drive our future research agenda. We first aim at deepening our analysis of how the research community has investigated non-functional requirements so far, possibly looking for scientific papers that, despite not explicitly targeting non-functional requirements, tangentially improve non-functional properties of machine learning-enabled systems. We will also work toward the definition of novel approaches that could optimize the non-functional requirements of machine learning-enabled systems while keeping accuracy into account. In addition, we aim to design software analytics studies, surveys and semi-structured interviews with machine learning engineers to further understand the impact of non-functional requirements and the practitioners' expectations with respect to them.

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: [60]

ACKNOWLEDGEMENT

The authors would like to thank the Associate Editor and Reviewers of TOSEM for their constructive and valuable inputs, which substantially improved the quality of this work. This work has been partially supported by the *Qual-AI* and *FRINGE* national research projects, which have been funded by the MUR under the PRIN 2022 and PRIN 2022 PNRR programs (Codes: D53D23008570006 and D53D23017340001, respectively).

REFERENCES

[1] 1994. UCI Statlog (German Credit Data) Data Set. https://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data)

[2] 1996. UCI Adult Data Set. https://archive.ics.uci.edu/ml/datasets/adult

[3] 2001. UCI heart disease data set. https://archive.ics.uci.edu/ml/datasets/Heart+Disease

[4] Alex Acero. 1992. *Acoustical and environmental robustness in automatic speech recognition*. Vol. 201. Springer Science & Business Media.

[5] Charu C Aggarwal et al. 2015. *Data mining: the textbook*. Vol. 1. Springer.

[6] 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* (2023), 107176.

[7] 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.

[8] 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.

[9] 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.

[10] 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

[11] 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.

[12] 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.

[13] 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).

[14] Pierre Bourque and E Richard. 2014. Swebok Version 3.0. *IEEE, ISBN-10: 0-7695-5166-1* (2014).

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

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

[17] 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.

[18] Victor R Basili1 Gianluigi Caldiera and H Dieter Rombach. 1994. The goal question metric approach. *Encyclopedia of software engineering* (1994), 528–532.

[19] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374* (2021).

[20] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190* (2018).

[21] E Dataset. 2011. Novel datasets for fine-grained image categorization. In *First Workshop on Fine Grained Visual Categorization, CVPR. Citeseer. Citeseer.* Citeseer.

[22] 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.

[23] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

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

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

[26] Andreas Eisele and Yu Chen. 2010. Multiun: A multilingual corpus from united nation documents.. In *LREC*.

[27] 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.

[28] Enrique Garcia-Ceja, Michael Riegler, Petter Jakobsen, Jim Tørresen, Tine Nordgreen, Ketil J. Oedegaard, and Ole Bernt Fasmer. 2018. Depresjon: A Motor Activity Database of Depression Episodes in Unipolar and Bipolar Patients. In *Proceedings of the 9th ACM Multimedia Systems Conference* (Amsterdam, Netherlands) *(MMSys '18)*. Association for Computing Machinery, New York, NY, USA, 472–477. https://doi.org/10.1145/3204949.3208125

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

[30] 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.

[31] 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

[32] 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.

[33] Khan Mohammad Habibullah and Jennifer Horkoff. 2021. Non-functional requirements for machine learning: understanding current use and challenges in industry. In *2021 IEEE 29th International Requirements Engineering Conference (RE)*. IEEE, 13–23.

[34] 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.

[35] François Hernandez, Vincent Nguyen, Sahar Ghannay, Natalia Tomashenko, and Yannick Estève. 2018. TED-LIUM 3: Twice as Much Data and Corpus Repartition for Experiments on Speaker Adaptation. In *Speech and Computer*, Alexey Karpov, Oliver Jokisch, and Rodmonga Potapova (Eds.). Springer International Publishing, Cham, 198–208.

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

[37] 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.

[38] 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

[39] Petter Jakobsen, Enrique Garcia-Ceja, Lena Antonsen Stabell, Ketil Joachim Oedegaard, Jan Oystein Berle, Vajira Thambawita, Steven Alexander Hicks, Pål Halvorsen, Ole Bernt Fasmer, and Michael Alexander Riegler. 2020. PSYKOSE: A Motor Activity Database of Patients with Schizophrenia. In *2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS)*. 303–308. https://doi.org/10.1109/CBMS49503.2020.00064

[40] 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.

[41] 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.

[42] 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.
[43] Barbara Kitchenham and Stuart Charters. 2007. Guidelines for performing Systematic Literature Reviews in Software Engineering. 2 (01 2007).
[44] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).
[45] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* 60, 6 (may 2017), 84–90. https://doi.org/10.1145/3065386
[46] Fumihiro Kumeno. 2019. Software engneering challenges for machine learning applications: A literature review. *Intelligent Decision Technologies* 13, 4 (2019), 463–476.
[47] 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.
[48] Yann LeCun. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/* (1998).
[49] 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).
[50] 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.
[51] 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.
[52] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.
[53] 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.
[54] 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).
[55] 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.
[56] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*. 142–150.
[57] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards Deep Learning Models Resistant to Adversarial Attacks. https://doi.org/10.48550/ARXIV.1706.06083
[58] Olvi L Mangasarian, W Nick Street, and William H Wolberg. 1995. Breast cancer diagnosis and prognosis via linear programming. *Operations research* 43, 4 (1995), 570–577.
[59] 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.
[60] Vincenzo De Martino and Fabio Palomba. 2023. Classification, Challenges, and Automated Approaches to Handle Non-Functional Requirements in ML-Enabled Systems: A Systematic Literature Review. (11 2023). https://doi.org/10.6084/m9.figshare.22815770.v3
[61] 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.
[62] 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.
[63] Tim Menzies and Thomas Zimmermann. 2013. Software analytics: so what? *IEEE Software* 30, 4 (2013), 31–37.
[64] Claire Cain Miller. 2015. Can an algorithm hire better than a human. *The New York Times* 25 (2015).
[65] Sérgio Moro, Paulo Cortez, and Paulo Rita. 2014. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems* 62 (2014), 22–31.
[66] 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.
[67] 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).

[68] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*. http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf
[69] MIT News. 2023. Shrinking deep learning's carbon footprint. https://news.mit.edu/2020/shrinking-deep-learning-carbon-footprint-0807. Accessed: 2023-11-26.
[70] Parmy Olson. 2011. The algorithm that beats your bank manager. *CNN Money March* 15 (2011).
[71] Abbas Ourmazd. 2020. Science in the age of machine learning. *Nature Reviews Physics* 2, 7 (2020), 342–343.
[72] Ariadna Quattoni and Antonio Torralba. 2009. Recognizing indoor scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 413–420. https://doi.org/10.1109/CVPR.2009.5206537
[73] 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.
[74] Jörg Rech and Klaus-Dieter Althoff. 2004. Artificial intelligence and software engineering: Status and future trends. *KI* 18, 3 (2004), 5–11.
[75] 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.
[76] 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.
[77] Anthony Rousseau, Paul Deléglise, Yannick Esteve, et al. 2014. Enhancing the TED-LIUM corpus with selected data for language modeling and more TED talks.. In *LREC*. 3935–3939.
[78] 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.
[79] Alex Serban and Joost Visser. 2021. An empirical study of software architecture for machine learning. *arXiv preprint arXiv:2105.12422* 39 (2021).
[80] 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.
[81] 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.
[82] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402* (2012).
[83] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. 2012. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks* 32 (2012), 323–332. https://doi.org/10.1016/j.neunet.2012.02.016 Selected Papers from IJCNN 2011.
[84] 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
[85] Yongqiang Tian, Zhihua Zeng, Ming Wen, Yepang Liu, Tzu-yang Kuo, and Shing-Chi Cheung. 2020. EvalDNN: A Toolbox for Evaluating Deep Neural Network Models. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Companion Proceedings* (Seoul, South Korea) *(ICSE '20)*. Association for Computing Machinery, New York, NY, USA, 45–48. https://doi.org/10.1145/3377812.3382133
[86] udacity challenge. 2016. Using Deep Learning to Predict Steering Angles. https://github.com/udacity/self-driving-car
[87] Andrew Varga and Herman J.M. Steeneken. 1993. Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Speech Communication* 12, 3 (1993), 247–251. https://doi.org/10.1016/0167-6393(93)90095-3
[88] 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
[89] Simin Wang, Liguo Huang, Amiao Gao, Jidong Ge, Tengfei Zhang, Haitao Feng, Ishna Satyarth, Ming Li, He Zhang, and Vincent Ng. 2022. Machine/deep learning for software engineering: A systematic literature review. *IEEE Transactions on Software Engineering* 49, 3 (2022), 1188–1231.
[90] 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).
[91] 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.
[92] Hironori Washizaki, Hiromu Uchida, Foutse Khomh, and Yann-Gaël Guéhéneuc. 2019. Studying software engineering patterns for designing machine learning systems. In *2019 10th International Workshop on Empirical Software Engineering in Practice (IWESEP)*. IEEE, 49–495.

[93] Cody Watson, Nathan Cooper, David Nader Palacio, Kevin Moran, and Denys Poshyvanyk. 2022. A systematic literature review on the use of deep learning in software engineering research. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 31, 2 (2022), 1–58.
[94] 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
[95] 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.
[96] 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.
[97] Lior Wolf, Tal Hassner, and Itay Maoz. 2011. Face recognition in unconstrained videos with matched background similarity. In *CVPR 2011*. 529–534. https://doi.org/10.1109/CVPR.2011.5995566
[98] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson, and Andreas Bulling. 2016. Learning an Appearance-Based Gaze Estimator from One Million Synthesised Images. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications* (Charleston, South Carolina) *(ETRA '16)*. Association for Computing Machinery, New York, NY, USA, 131–138. https://doi.org/10.1145/2857491.2857492
[99] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. 2018. MoleculeNet: a benchmark for molecular machine learning. *Chemical science* 9, 2 (2018), 513–530.
[100] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).
[101] 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
[102] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei. 2011. Human action recognition by learning bases of action attributes and parts. In *2011 International Conference on Computer Vision*. 1331–1338. https://doi.org/10.1109/ICCV.2011.6126386
[103] 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.
[104] 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.
[105] 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.
[106] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems* 28 (2015).
[107] Jianlong Zhou and Fang Chen. 2018. *Human and Machine Learning*. Springer.

SLR – SISTEMATIC LITERATURE REVIEW

[S1] 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.
[S2] 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.
[S3] 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.
[S4] 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.
[S5] 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.
[S6] 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.
[S7] 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.
[S8] 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.
[S9] 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.
[S10] Tejas S. Borkar and Lina J. Karam. Deepcorrect: Correcting dnn models against image distortions. *IEEE Transactions on Image Processing*, 28(12):6022–6034, 2019.
[S11] 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.
[S12] 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.
[S13] 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.
[S14] 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.
[S15] 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.
[S16] 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.
[S17] 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.
[S18] 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.
[S19] 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.
[S20] 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.
[S21] 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.
[S22] 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.
[S23] 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.
[S24] 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.

[S25] 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.

[S26] 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.

[S27] 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.

[S28] 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.

[S29] 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.

[S30] 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.

[S31] 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.

[S32] 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.

[S33] 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.

[S34] 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.

[S35] 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.

[S36] 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.

[S37] 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.

[S38] 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.

[S39] 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.

[S40] 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.

[S41] 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.

[S42] 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.

[S43] 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.

[S44] 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.

[S45] 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.

[S46] 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.

[S47] 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.

[S48] 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.

[S49] 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.

[S50] 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.

[S51] 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.

[S52] 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.

[S53] 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.

[S54] 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.

[S55] 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.

[S56] 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.

[S57] 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.

[S58] 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.

[S59] 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.

[S60] 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.

[S61] 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.

[S62] 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.

[S63] 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.

[S64] 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.

[S65] 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.

[S66] 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.

[S67] 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.

[S68] 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.

[S69] 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.