



Software Engineering for AI-Based Systems: A Survey

SILVERIO MARTÍNEZ-FERNÁNDEZ, Universitat Politècnica de Catalunya - BarcelonaTech, Spain

JUSTUS BOGNER, University of Stuttgart, Institute of Software Engineering, Germany

XAVIER FRANCH and MARC ORIOL, Universitat Politècnica de Catalunya - BarcelonaTech, Spain

JULIEN SIEBERT, ADAM TRENDOWICZ, and ANNA MARIA VOLLMER, Fraunhofer

Institute for Experimental Software Engineering IESE, Germany

STEFAN WAGNER, University of Stuttgart, Institute of Software Engineering, Germany

AI-based systems are software systems with functionalities enabled by at least one AI component (e.g., for image-, speech-recognition, and autonomous driving). AI-based systems are becoming pervasive in society due to advances in AI. However, there is limited synthesized knowledge on **Software Engineering (SE)** approaches for building, operating, and maintaining AI-based systems. To collect and analyze state-of-the-art knowledge about SE for AI-based systems, we conducted a systematic mapping study. We considered 248 studies published between January 2010 and March 2020. SE for AI-based systems is an emerging research area, where more than 2/3 of the studies have been published since 2018. The most studied properties of AI-based systems are dependability and safety. We identified multiple SE approaches for AI-based systems, which we classified according to the SWEBOK areas. Studies related to software testing and software quality are very prevalent, while areas like software maintenance seem neglected. Data-related issues are the most recurrent challenges. Our results are valuable for: researchers, to quickly understand the state-of-the-art and learn which topics need more research; practitioners, to learn about the approaches and challenges that SE entails for AI-based systems; and, educators, to bridge the gap among SE and AI in their curricula.

CCS Concepts: • Software and its engineering → Software creation and management; • Computing methodologies → Machine learning;

Additional Key Words and Phrases: Software engineering, artificial intelligence, AI-based systems, systematic mapping study

ACM Reference format:

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 Trans. Softw. Eng. Methodol.* 31, 2, Article 37e (March 2022), 59 pages.

<https://doi.org/10.1145/3487043>

This work has been partially funded by the “Beatriz Galindo” Spanish Program BEAGAL18/00064 and by the DOGO4ML Spanish research project (ref. PID2020-117191RB-I00). We are very grateful to our anonymous reviewers for their comments and suggestions.

Authors' addresses: S. Martínez-Fernández, X. Franch, and M. Oriol, Universitat Politècnica de Catalunya - BarcelonaTech, c/Jordi Girona 1-3, 08034 Barcelona (Spain); emails: silverio.martinez@upc.edu, franch@essi.upc.edu, moriol@essi.upc.edu; J. Bogner and S. Wagner, University of Stuttgart, Institute of Software Engineering, Germany, Universitätsstraße 38, 70569 Stuttgart (Germany); emails: justus.bogner@iste.uni-stuttgart.de, stefan.wagner@iste.uni-stuttgart.de; J. Siebert, A. Trendowicz, and A. M. Vollmer, Fraunhofer Institute for Experimental Software Engineering IESE, Fraunhofer-Platz 1, 67663 Kaiserslautern (Germany); emails: julien.siebert@iese.fraunhofer.de, adam.trendowicz@iese.fraunhofer.de, anna-maria.vollmer@iese.fraunhofer.de.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

© 2022 Copyright held by the owner/author(s).

1049-331X/2022/03-ART37e \$15.00

<https://doi.org/10.1145/3487043>

1 INTRODUCTION

In the last decade, increased computer processing power, larger datasets, and better algorithms have enabled advances in **Artificial Intelligence (AI)** [11]. Indeed, AI has evolved towards a new wave, which Deng calls “the rising wave of **Deep Learning**” (**DL**) [46].¹ DL has become feasible, leading to **Machine Learning (ML)** becoming integral to many widely used software services and applications [46]. For instance, AI has brought a number of important applications, such as image- and speech-recognition and autonomous, vehicle navigation, to near-human levels of performance [11].

The new wave of AI has hit the software industry with the proliferation of AI-based systems integrating AI capabilities based on advances in ML and DL [6, 24]. AI-based systems are software systems which include AI components. These systems learn by analyzing their environment and taking actions, aiming at having an intelligent behaviour. As defined by the expert group on AI of the European Commission, “AI-based systems can be purely software-based, acting in the virtual world (e.g., voice assistants, image analysis software, search engines, and speech and face recognition systems) or AI can be embedded in hardware devices (e.g., advanced robots, autonomous cars, drones, or Internet of Things applications)”.²

Building, operating, and maintaining AI-based systems is different from developing and maintaining traditional software systems. In AI-based systems, rules and system behaviour are inferred from training data, rather than written down as program code [101]. AI-based systems require interdisciplinary collaborative teams of data scientists and software engineers [6]. The quality attributes for which we need to design and analyze are different [153]. The evolution of AI-based systems requires focusing on large and changing datasets, robust and evolutionary infrastructure, and ethics and equity requirements engineering [119]. Without acknowledging these differences, we may end up creating poor AI-based systems with technical debt [179].

In this context, there is a need to explore **Software Engineering (SE)** practices to develop, maintain, and evolve AI-based systems. This paper aims to characterize SE practices for AI-based systems in the new wave of AI, i.e., **Software Engineering for Artificial Intelligence (SE4AI)**. The motivation of this work is to synthesize the current SE knowledge pertinent to AI-based systems for: researchers to quickly understand the state of the art and learn which topics need more research; practitioners to learn about the approaches and challenges that SE entails when applied to AI-based systems; and educators to bridge the gap among SE and AI in their curricula.

Bearing this goal in mind, we have conducted a **Systematic Mapping Study (SMS)** considering literature from January 2010 to March 2020. The reason to focus on the last decade is that this new wave of AI started in 2010, with industrial applications of DL for large-scale speech recognition, computer vision and machine translation [46].

The main contributions of this work are the synthesis of:

- Bibliometrics of the state of the art in SE4AI (see Section 4).
- Characteristics of AI-based systems, namely scope, application domain, AI technology, and key quality attribute goals (see Section 5).
- SE approaches for AI-based systems following the Knowledge Areas of SWEBOk, a guide to the SE Body of Knowledge [25] (see Section 6).
- Challenges of SE approaches for AI-based systems following SWEBOk Knowledge Areas (see Section 7).

¹https://en.wikipedia.org/wiki/History_of_artificial_intelligence#Deep_learning,_big_data_and_artificial_general_intelligence:_2011-present.

²<https://ec.europa.eu/digital-single-market/en/news/definition-artificial-intelligence-main-capabilities-and-scientific-disciplines>.

2 BACKGROUND

This section respectively discusses the synergies between SE and AI, and related work.

2.1 On the Synergies between SE and AI

History of AI and SE. AI and SE are naturally related, as they both have their roots in computer science. Although AI as we know it today may be traced back to the early 50s or 40s, software development and AI met for the first time when the Mark 1 perceptron was programmed on an IBM 704 machine for image recognition, in 1959 – although the perceptron itself was initially intended to be a machine and software development has rather little to do with SE as we know it today. SE as a profession and research area was developed in the late 1960s [148], when the term “software crisis” was coined relating to problems in engineering’s increasingly large and complex software systems. The boom of Expert Systems in the 80s brought AI back to the forefront again [169]. The deployment and use of expert systems in production systems has naturally led to a series of SE challenges (like validation & verification). For example, Partridge [154] surveyed the relationships between AI and SE. Despite addressing problems of a different nature, AI and SE have been intertwined since their very beginnings: AI methods have been used to support SE tasks (AI4SE) and SE methods have been used to develop AI (SE4AI) software.

AI4SE. Recently, Perkusich et al. [157] referred to AI4SE as intelligent SE and defined it as a portfolio of SE techniques, which “explore data (from digital artifacts or domain experts) for knowledge discovery, reasoning, learning, planning, natural language processing, perception or supporting decision-making”. AI4SE has developed driven by the rapid increase in size and complexity of software systems and, in consequence, of SE tasks. Wherever software engineers came to their cognitive limits, automatable methods were the subject of research. While searching for solutions, the SE community observed that a number of SE tasks can be formulated as data analysis (learning) tasks and thus can be supported, for example, with ML algorithms.

SE4AI. First applications of SE to AI were limited to simply implementing AI algorithms as standalone programs, such as the aforementioned Mark 1 perceptron. As AI-based software systems grew in size and complexity, and as its practical and commercial application increased, more advanced SE methods were required. The breakthrough took place when AI components became a part of established software systems, such as expert systems, or driving control. It quickly became clear that, because of the specific nature of AI (e.g., dependency on learning data), traditional SE methods were not suitable anymore (e.g., leading to technical debt [179]). This called for revision of classical, and development of new, SE paradigms and methods. This paper provides a comprehensive review of what has been achieved in the area so far.

2.2 Related work on SE4AI

Several secondary studies in the broad area of SE4AI have been published so far (see Table 1).

Masuda et al. [134] conducted a review to identify techniques for the evaluation and improvement of the software quality of ML applications. They analyzed 101 papers and concluded that the field is still in an early state, especially for quality attributes other than functional correctness and safety.

Washizaki et al. [211] conducted a multivocal review to identify architecture and design patterns for ML systems. From 35 resources (both white and grey literature), they extracted 33 unique patterns and mapped them to the different ML phases. They discovered that, for many phases, only very few or even no patterns have been conceptualized so far. Serban and Visser [181] performed both a case study and a systematic literature review on the topic of software architecture

for machine learning. They reviewed 42 studies and performed 10 semi-structured interviews with practitioners from 10 different organisations. In their paper, the authors report 20 challenges and potential solutions. On a similar topic, John et al. [137], performed a systematic review of both scientific (13 studies) and grey literature (6 studies) on the topic of deployment of ML systems. They report a total of 27 challenges and 52 practices. Lorenzoni et al. [124] performed a systematic literature review on the topic of development of machine learning systems. They analysed 33 studies between 2010 and 2020 and classified 10 issues and 13 solutions into seven SE practices.

A number of reviews have been conducted in the area of software testing. Borg et al. [23] performed a review of verification and validation techniques for **deep neural networks (DNNs)** in the automotive industry. From 64 papers, they extracted challenges and verified them with workshops and finally a questionnaire survey with 49 practitioners. They conclude, among other challenges, that a considerable gap exists between safety standards and nature of contemporary ML-based systems. Another study was published by Ben Braiek and Khomh [27]. In their review of testing practices for ML programs, they selected a total of 37 primary studies and extracted challenges, solutions, and gaps. The primary studies were assigned to the categories of detect errors in data (5 papers), in ML models (19 papers), and in the training program (13 papers). Riccio et al. [166] extracted testing challenges from 70 primary studies and propose the following categories: realism of test input data (5 papers), adequacy of test criteria (12 papers), identification of behavioural boundaries (2 papers), scenario specification and design (3 papers), oracle (13 papers), faults and debugging (8 papers), regression testing (5 papers), online monitoring and validation (8 papers), cost of testing (10 papers), integration of ML models (2 papers), and data quality assessment (2 papers). Similarly, Zhang et al. [226] surveyed the literature on ML testing and selected 138 papers. From these, they summarized the tested quality attributes (e.g., correctness or fairness), the tested components (e.g., the data or the learning program), workflow aspects (e.g. test generation or evaluation), and application contexts (e.g., autonomous driving or machine translation).

In addition to these specialized reviews, there are also some secondary studies more similar to ours, i.e., that have a general SE focus. Serban et al. [180] conducted a multivocal review with 21 relevant documents (both white and grey literature) to identify and analyze SE best practices for ML applications. They extracted 29 best practices and used a follow-up questionnaire survey with 313 software professionals to find out the degree of adoption and impact of these practices. Furthermore, Wang et al. [208] took a broader view and conducted a systematic literature review about general synergies between ML/DL and SE, i.e., covering both machine learning for SE (ML4SE) and **SE for machine learning (SE4ML)** research. However, only 15 of the 906 identified studies covered the SE4ML direction. Based on their results, the authors concluded that “it remains difficult to apply SE practices to develop ML/DL systems”. Another systematic literature review was performed by Kumeno [115]. He focused solely on the extraction of SE challenges for ML applications and mapped them to the different SWEBOK areas. In total, he selected 115 papers (47 papers focusing on SE-related challenges for ML and 68 papers focusing on ML-techniques or ML-applications challenges) from 2000 to 2019. Moreover, Lwakatare et al. [126] conducted a similar review of SE challenges faced by industrial practitioners in the context of large-scale ML systems. They categorized 23 challenges found in the 72 papers selected according to four quality attributes (adaptability, scalability, privacy, safety) and four ML process stages (data acquisition, training, evaluation, deployment). Adaptability and scalability were reported to face a significantly larger number of challenges than privacy and safety. They also identified eight solutions, e.g., transfer learning and synthetically generated data, solving up to 13 of the challenges, especially adaptability. Giray [71] also conducted a systematic literature review to identify the state of the art and challenges in the area of ML systems engineering. In his sampling method, he exclusively targeted publications from SE venues and selected 141 studies. These studies were then analyzed for their bibliometrics,

the used research methods, plus mentioned challenges and proposed solutions. Lastly, Nascimento et al. [147] performed an SLR to analyze how SE practices have been applied to develop AI or ML systems, with special emphasis on limitations and open challenges. They also focus on system contexts, challenges, and SE contribution types. While they considered publications between 1990 and 2019, they only selected 55 papers.

We summarize in Table 1 the findings of the related work. This summary has a three-fold objective: (i) to provide a synthetic view of the approaches aforementioned; (ii) to make evident our claim that no systematic mapping or general review with a breadth and depth similar to ours has been published so far; and (iii) to facilitate the comparison of the results of our study with the related work.

In summary, even though several secondary studies have recently been published or submitted (a few of the aforementioned studies are pre-prints), there are still very few works that broadly summarize and classify research in the field of SE for AI-based systems. No systematic mapping or general review with a breadth and depth similar to ours has been published so far. Existing general reviews focus either exclusively on challenges, analyze considerably fewer studies, or only take publications with an industry context into account, i.e., they partially fail to describe the wide spectrum of results in this research area.

The main objective of our study is therefore to systematically select and review the broad body of literature and to present a holistic overview of existing studies in SE for AI-based systems. An SMS is a proven and established method to construct such a holistic overview.

3 RESEARCH METHODOLOGY

This SMS has been developed following the guidelines for performing SMSs from Petersen et al. [159]. Additionally, the guidelines for systematic literature reviews provided by Kitchenham and Charters [104] have also been used when complementary. This is because the method for searching for the primary studies, and taking a decision about their inclusion or exclusion is very similar between a systematic literature review and an SMS. This SMS has five main steps [112, 158] described in the following subsections.

3.1 Definition of Research Questions

The aim of this SMS consists of identifying the existing research of SE for AI-based systems by systematically selecting and reviewing published literature, structuring this field of interest in a broader, quantified manner. We define the goal of this SMS formally using GQM [17]: *Analyze SE approaches for the purpose of structure and classification with respect to proposed solutions and existing challenges from the viewpoint of both SE researchers and practitioners in the context of AI-based systems in the new wave of AI.*

To further refine this objective, we formulated four **Research Questions (RQs)** to be answered by our primary studies:

- RQ1. How is SE research for AI-based systems characterized?
- RQ2. What are the characteristics of AI-based systems (used terms, scope, and quality goals)?
- RQ3. Which SE approaches for AI-based systems have been reported in the scientific literature?
- RQ4. What are the existing challenges associated with SE for AI-based systems?

The research methodology followed to answer each of these RQs is detailed in Section 3.5.

3.2 Conduct Search

Since AI-based systems cover interdisciplinary communities (SE and AI), it is difficult to find an effective search string. Therefore, we decided to execute a hybrid search strategy [141], applying

Table 1. Summary of Relevant Aspects on Bibliometrics, AI-based System Properties, SE Approaches, and Challenges Found in Related Work

Study	Bibliometrics	AI-based systems properties	SE approaches	Challenges
[134]	101 studies (2005–2018)	safety, correctness	practices for the evaluation and improvement of the software quality of ML applications	ML quality assurance
[211]	38 studies (2008–2019)		33 unique architecture and design patterns for ML systems	
[23]	64 studies (2002–2007, 2013–2016)	safety, robustness, reliability	verification and validation techniques for safety-critical automotive systems	verification and validation in DNNs
[27]	37 studies (2012–2018)	correctness	testing practices	18 challenges organized in 6 dimensions: implementation issues, data issues, model issues, written code issues, execution environment issues, mathematical design issues
[166]	70 studies (2004–2019)	fairness, accuracy, safety, consistency	functional testing, test case generation and test oracle, integration testing, system testing	11 challenges
[226]	138 studies (2007–2019)	correctness, model relevance, robustness, security, data privacy, efficiency, fairness, interpretability	testing workflow, testing components, testing properties, application scenarios	4 testing challenges categories: test input generation, test assessment criteria, oracle problem, testing cost reduction
[180]	21 studies (2017–2019)		29 best practices for ML systems in six categories: data, training, coding, deployment, team, governance	
[208]	15 studies (out of 906) (2009–2018)		Model Evaluation, Deployment	
[115]	115 studies (2003–2019)	safety, security, ethics and regulation, software structure, testability, maintainability, performance, risk and uncertainty, economic impact	all SWEBOk areas	SE challenges for ML applications
[126]	72 studies (1998–2018)	adaptability, scalability, safety, privacy	Software construction. Software maintenance	23 SE challenges faced by practitioners in the context of large-scale ML systems. 13 of them had solutions
[71]	141 studies (2007–2019)		requirements engineering, design, software development and tools, testing and quality, maintenance and configuration management, SE process and management, organizational aspects	31 challenges and partial solutions that have been raised by SE researchers
[147]	55 studies (1999–2019)	AI ethics, interpretability, scalability, explanation on failure, complexity, efficiency, fairness, imperfection, privacy, safety, safety and stability, robustness, reusability, stability, staleness	AI software quality, data management, project management, infrastructure, testing, model development, requirement engineering, AI engineering, architecture design, model deployment, integration, education, operation support.	SE challenges for ML applications and how SE practices adjusted to deal with them.
[124]	33 studies (2010–2020)		data processing, documentation and versioning, non-functional requirements, design and implementation, evaluation, deployment and maintenance, software capability maturity model	10 issues and 13 solutions
[181]	42 studies (2016–2021)	Performance, scalability, interpretability, hardware resources, interoperability, robustness, generalization, low data quality, scarcity of data, maintainability, privacy, security	Requirements, data, design, testing, operations, organisation	20 challenges and potential solutions
[137]	19 studies (2017–2020)		Design, integration, deployment, operation, evolution.	27 challenges and 52 practices
This study	248 studies (2010–2020)	40 quality attributes (see sections 4 and 5)	11 SWEBOk areas (see Section 6)	94 challenges (see Section 7)

both a search string in Scopus³ and a snowballing strategy [212]. This strategy is referred to as “Scopus + BS*FS” [141]: it first runs a search over Scopus to get a start set of papers and compose a seed set for snowballing. The Scopus database contains peer-reviewed publications from top SE journals and conferences, including IEEE Xplore, ACM Digital library, ScienceDirect (Elsevier), and Springer research papers. We have recently assessed that Scopus’ coverage is optimal when compared to these other databases [41]. Scopus enabled us to search with one engine for relevant studies considering a large library content and using different search engine functionalities such as export of the results and formulation of own search strings. We iteratively created several initial search strings, discussed their containing terms, and compared the results. The final search string used on Scopus was:

```

TITLE-ABS-KEY("software engineering for artificial intelligence" OR
"software engineering for machine learning" OR
(("software engineering" OR "software requirement*" OR "requirements engineering" OR
"software design" OR "software architecture" OR "software construction" OR "software testing"
OR "software maintenance" OR "software configuration management" OR "software quality")
AND
("AI-enabled system*" OR "AI-based system*" OR "AI-infused system*" OR "AI software" OR
"artificial intelligence-enabled system*" OR "artificial intelligence-based system*" OR "artificial
intelligence-infused system*" OR "artificial intelligence software" OR "ML-enabled system*"
OR "ML-based system*" OR "ML-infused system*" OR "ML software" OR "machine
learning-enabled system*" OR "machine learning-based system*" OR "machine learning-
infused system*" OR "machine learning software"))
)
```

With this search string, we aimed to capture the literature on SE4AI. Since this is a concept recently popular in the SE community, it was directly used as part of the search string. Furthermore, we also included subareas of AI, such as ML. Finally, the search string also considered the combination of SE Knowledge Areas from SWEBOK v3.0 [25] and many similar terms to AI-based systems. We ran a check whether this search string included key papers on the topic of SE4AI from an annotated bibliography of an (external) expert on the topic,⁴ and many were included. Therefore, we considered it as a good starting point to be complemented with other techniques to mitigate selection bias.

An important decision was to only consider primary studies belonging to the new wave of AI from January 2010 to March 2020. The reason is that we wanted to structure the knowledge of SE4AI since the new wave of AI [11, 46]. We applied the search string on Scopus to this interval of time on April 2nd, 2020, which resulted in 116 studies (see Figure 1).

Then, other papers were obtained from the seed set via backward and forward snowballing. We applied snowballing as indicated in the guidelines by Wohlin [212]. The index used to check the number of references was Google Scholar. While we considered all the citations of each paper during backward snowballing, we established a limit of the first 100 citations returned by Google Scholar during forward snowballing. Each of the authors checked these references and citations of randomly assigned papers. To avoid missing relevant papers, if there was any hint that a study could be included, it was inserted in our snowballing working document.

In total, we performed two snowballing iterations as reported in this study. For each entry in the aforementioned snowballing working document, at least two researchers applied inclusion and exclusion criteria. The next subsection explains this selection process of screening papers.

³<https://www.scopus.com>.

⁴<https://github.com/ckaestne/seaibib>.

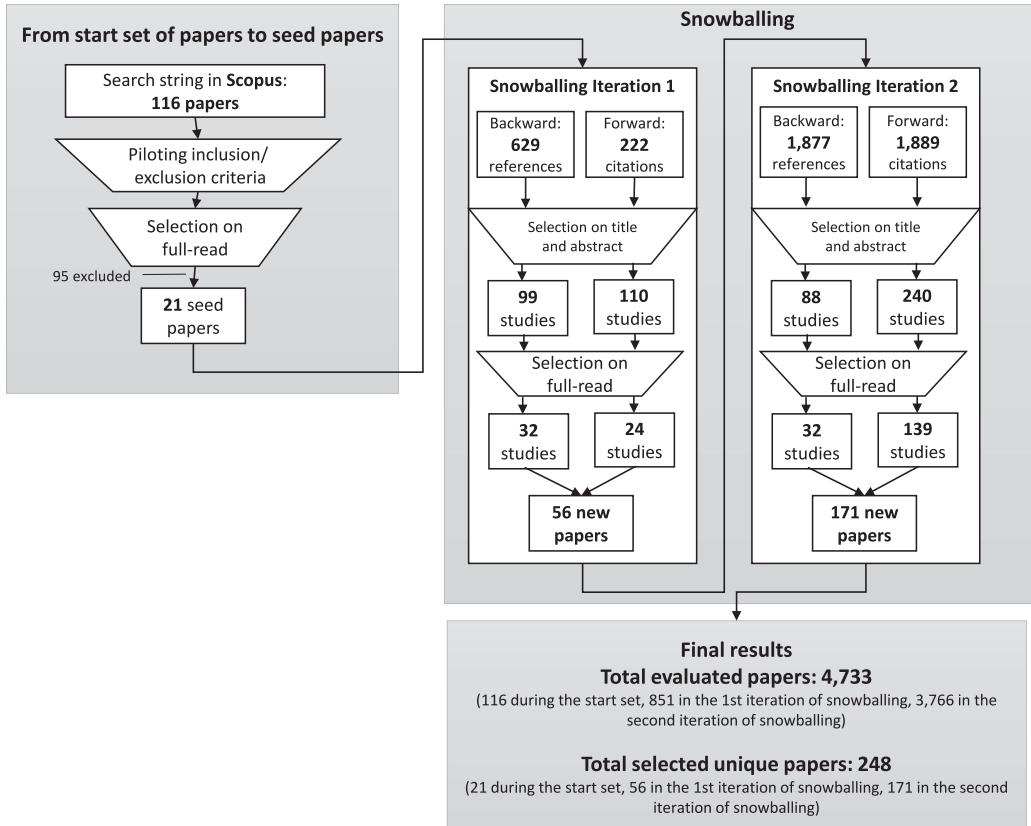


Fig. 1. Search process (selection on title and abstract appear in the snowballing file, new papers are in the data extraction).

3.3 Screening of Papers

Based on our RQs, we specified the inclusion and exclusion criteria listed in Table 2. This supported a consistent screening among all of us. These criteria had been used to select relevant papers first based on their titles and abstracts and afterward based on the full-reads. We received English titles and abstracts for all the obtained studies, so there are no exclusions based on the language of the primary study. Knowledge of the authors, institutions, journals, and year of publication had not been removed during the study selection process. Evidence suggests that masking the origin of primary studies does not improve reviews [103]. As we started the search for primary studies beginning of April 2020, we excluded papers that were published after March 2020 but identified through forward snowballing.

For screening each paper identified both in the search in Scopus and the snowballing, we assigned two researchers who checked the abstract and title independently of each other. We made sure to mix the assignments of persons so that each person had a similar amount of primary studies with everybody from the research team, and permuting the assignments, so that pairs were balanced. In case of disagreement, a third person assisted to find a common decision. This aimed to improve the reliability of our systematic mapping [213]. The inter-rater agreement before the third person was involved was 0.751 using Cohen's kappa coefficient, which indicates a substantial agreement among participants [53]. The full-read was done by one person and if the final

Table 2. Inclusion and Exclusion Criteria

Criteria	Description
Inclusion (IC1)	The work is mainly focused on SE for AI-based systems.
Exclusion (EC1)	The work does not fulfill IC1 (e.g., is focused on AI technologies for SE).
Exclusion (EC2)	The work is not accessible even after contacting the authors.
Exclusion (EC3)	The work is not completely written in English.
Exclusion (EC4)	The work is a secondary study (e.g., SLR).
Exclusion (EC5)	The work is an exact duplicate of another study.
Exclusion (EC6)	The work is a short paper of two pages or less.
Exclusion (EC7)	The work is not a research paper published in books, journals, conferences, workshops, or the arXiv repository (e.g., an editorial for a special issue, a table of contents of proceedings, short course description, tutorial, summary of a conference, Ph.D. thesis, master thesis, blog, technical report).
Exclusion (EC8)	The work has been published before January 2010 or after March 2020.

decision mismatched the previous agreement, all included persons were informed to eventually decide about the inclusion or exclusion of the corresponding paper. We documented these results in our snowballing working document to ease the transparency among us. Following this, we identified in total 21 relevant seed papers, 56 additional papers based on the first snowballing iteration, and additional 171 papers in our second snowballing iteration as illustrated in Figure 1. In 53 cases (9.83%), a third person was needed to resolve the disagreements. After finishing the second snowballing iteration, we included 248 primary studies in total. As the field of SE4AI is further emerging and our resources for this study are limited, we decided to stop at this point.

3.4 Keywording

We created a data extraction form containing four different kinds of information to gain a broad view of SE for AI-based systems related to our RQs. We planned to collect the following data:

- (1) Generic data and bibliometrics: contains demographic information as well as the used criteria to assess the quality of our selected studies.
- (2) AI-related terms: includes terms for the targeted AI-based system or subsystem/component of the AI-based system as used in the primary studies, addressed AI-based system properties (e.g., explainability, safety, etc.) as focused by the primary studies, definitions of the terms of the targeted AI-based system, and domain names as reported in the primary studies.
- (3) SE approaches for AI-based systems: contains the most suitable Knowledge Areas of the reported SE approaches for AI-based systems, the SE approach name or description categorized according to different types of approaches (method, model, practice, tool, framework, guideline, other), and the observed impact of applying the SE approach.
- (4) Challenges of SE approaches for AI-based systems: list of explicitly stated challenges reported in the primary studies and, if available, possible solutions and relevant quotes regarding SE approaches for AI-based systems.

For our data extraction form, we made use of existing classification criteria coming from the SE domain. More precisely, we applied Ivarsson and Gorscheck's rigor and relevance quality assessment model [93] for the quality appraisal (see Section 4.2), and we used the Knowledge Areas listed in the SWEBOK v3.0 [25] to classify the identified SE approaches for AI-based systems. These are the corresponding Knowledge Areas: Software Requirements, Software Design, Software Construction, Software Testing, Software Maintenance, Software Configuration Management,

Software Engineering Management, Software Engineering Process, Software Engineering Models and Methods, Software Quality, Software Engineering Professional Practice, Software Engineering Economics, Computing Foundations, Mathematical Foundations, and Engineering Foundations.

The values of some criteria were fixed in advance (e.g., the aforementioned quality appraisal criteria values and the predefined Knowledge Areas), whilst others emerged in the analysis phase after reading the papers (e.g., AI-related terms, focused quality properties).

We piloted the initial data extraction form to both ensure a common understanding among all the involved researchers and extend the form if required, especially with additional values for the classification criteria. For this piloting activity, we randomly selected three of our seed papers, fully read the papers, extracted them independently by the eight authors, and discussed our results together. As a consequence, we improved the extraction form to make a few columns more objective and decided to use literal quotations for some columns (e.g., challenges). Because we expected that the topic of our study is quite diverse, we also decided to perform the data extraction based on full-read. Just checking the abstracts, introductions, and conclusions would not contain enough information to create the final keywording, i.e., the classification criteria with several values. Furthermore, other researchers reported that a quick keywording relies on good quality of the abstracts, and in the SE domain they may not be enough to conduct the keywording [30, 158].

During the discussions of our data extraction form, we observed that the used quality appraisal classification criteria by Ivarsson and Gorscheck are missing specific values for better capturing the AI-based systems or the study types used in the AI/ML community. For example, we identified several primary studies that compared the performance and outcome of an AI/ML system with other systems or under different environments. As we count this as a type of empirical study, we added a new method type named *Benchmark*. Primary studies belong to this kind of study type if they describe a rigorous study that evaluates one or more algorithms in well-established operational settings (with data and variables). Furthermore, we extended the definitions of the criteria *Industrial Relevance* by including AI/ML algorithms, systems, and data sets as another type of subject. This allowed us to assess the industrial relevance of Benchmark studies.

3.5 Data Extraction and Mapping Process

Each of the 248 primary studies was assigned to a single researcher for extraction based on the predefined data extraction form. Extractors were in frequent asynchronous contact to discuss potential inconsistencies. The weekly project meeting was also used for synchronization on this matter. In these meetings, we discussed the most persistent conflicts and shared our way of working as well as emerging findings to ensure cohesion of the team and to minimize subjectivity. Additionally, the data collection form includes the name of the reviewer and space for additional notes. This enabled us also to keep track of this process.

After completing the extraction, we started data analysis and synthesis. In general, we performed both quantitative and qualitative analyses to classify the extracted data. During this process, additional extraction inconsistencies and mistakes were discovered and easily resolved in direct communication with the original extractors. Synthesis per RQ was performed by groups of at least two researchers, who often re-read (parts of) the original papers for this and kept the group in the loop. Final results were presented to the rest of the team and feedback was incorporated.

Regarding the required mapping and analysis to answer the RQs, we performed the following activities in each RQ:

For the analysis of RQ1, we used the assessed rigor and relevance quality [93] and performed frequency analysis to additionally determine bibliographical data such as the annual publication trend, venue types, authors' affiliations, geography distribution, and the empirical research type of the primary studies.

To answer RQ2, we counted the number of occurrences of various terms related to AI used to characterize the study objects. We then used inductive coding to distill dimensions (Scope, Application Domain, and Technologies of AI) to describe the study objects, coded all primary studies, and reported frequencies. For the key quality attribute goals of AI-based systems, this also included a harmonization of the used terms as well as axial coding to cluster the identified quality properties.

As RQ3 and RQ4 highly rely on extensive qualitative analysis, all eight authors supported the analysis. Therefore, we split the primary studies according to the extracted SWEBOK Knowledge Areas and equally distributed the number of studies to groups of two or three researchers. Again, we performed thematic analysis [43] and derived several codes independently for the different Knowledge Areas (RQ3) and uniformed them to our subcategories for each of the SWEBOK Knowledge Areas. Finally, we mapped the primary studies accordingly, whereby one primary study could belong to several Knowledge Areas. For each Knowledge Area, each primary study could also be mapped to several subcategories, but required at least one.

As we created a subcategory called “challenges” for each Knowledge Area, we restricted the analysis for RQ4 regarding challenges in SE4AI explicitly to those primary studies whose classification under RQ3 included a mapping to this subcategory. For the mapping of the challenges, we made use of the different Topics provided for each Knowledge Area by SWEBOK. We mapped a challenge into one or more SWEBOK Topics if the link was evident. Moreover, we added information about root causes, mitigation actions, and impact on the data extraction form as a basis for some additional qualitative analysis. Finally, we uniformed the terminology (near 80% of the Topics were rewritten to some extent) and the classification (changes of SWEBOK Topic or even Knowledge Area, assignment of an additional Topic to a challenge, or removal of a Topic). It also resulted in removing some challenges because they are duplicated in another primary study with proper citation (e.g., [136] presenting the same four challenges as [7]) or are too generic to allow proper analysis. The three researchers responsible for RQ4 decided to apply inductive coding and added a new Knowledge Area called *Software Runtime Behaviour* that they considered not well-covered in SWEBOK. Our new Knowledge Area was finally composed of three Topics: *Cost & Efficiency*, *Quality*, and *Robustness & Operability*. Furthermore, seeking conceptual consistency inside some SWEBOK Topics, we proposed the following new Topics:

- *ML/AI methods*, in SE Models and Methods Knowledge Area. This new Topic could be numbered as 9.4.5, i.e., at the same level as existing particular types of methods, e.g., Formal Methods (9.4.2) or Agile Methods (9.4.4).
- *ML/AI specific activities*, in Software Life Cycles Topic (8.4.2). The reason is that while these activities are part of ML/AI-based software development, the current formulation of the Topic does not explicitly address the definition of concrete activities in the life cycle.
- *Data-related issues* and *Process-related issues*, as sub-Topics in Software Testing: Key Issues (4.1.2). We considered it convenient to clearly distinguish among these two types of issues related to testing due to their different nature.
- *Applicability and adoption of research results in practice*, as sub-Topic of SE Professional Practice. We thought that this important aspect is not covered by any of the three main Topics in the Knowledge Area (Professionalism, Group Dynamics and Psychology, and Communication Skills). Therefore, this proposed Topic could be numbered as 11.4.

In the replication package⁵, the reader can see: the results of the search on Scopus for seed papers (and alternative discarded search procedures), the result of the application of inclusion and

⁵Replication package available on <https://doi.org/10.6084/m9.figshare.14538324>.

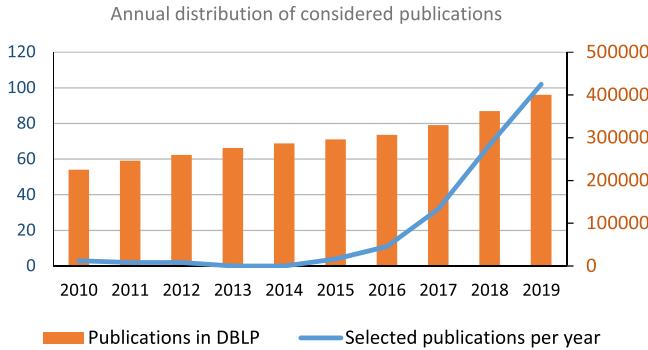


Fig. 2. Annual distribution of publications.

exclusion criteria for the papers found, the document used for snowballing, the data extraction form, and analysis files for each RQ.

4 RQ1: HOW IS SOFTWARE ENGINEERING RESEARCH FOR AI-BASED SYSTEMS CHARACTERIZED?

This section respectively discusses the bibliometrics of the primary studies, and their research rigor and industrial relevance.

4.1 Bibliometrics

We focus the bibliometric analysis into four aspects: annual trend, distribution per venue type, distribution per affiliation type, and geographical distribution.

Annual trend. We observe a first period in which the number of publications were marginal (2010–2012) or even nonexistent (2013–2014) (see Figure 2). But since 2015, we observe a rapidly increasing growth in the field, with the number of publications being approximately doubled every year since (from 4 in 2015 to 102 in 2019). This trend is well above the overall trend of increasing number of publications in DBLP, shown in the same figure for comparison.

The numbers for the year 2020 are not shown, as this year is not completely covered in the study (the timespan of the study covers until March 2020).

Distribution per venue type. As shown in Figure 3, most of the primary studies were published in conferences, and remarkably, there is also a considerable number of publications which were published as arXiv reports.

Distribution per affiliation type. Figure 4 depicts the type of affiliation of the authors of the research papers analysed. Academia includes authors with affiliation in research centers.

Geographic distribution. We examined the affiliation country of the first author of the papers (see Figure 5). As shown, the USA plays a leading role in 40% of the studies analysed, which multiplies China's leading role by a 4x factor and Germany and Japan by 5x. If we observe the distribution by continents, North America doubles Asia in absolute numbers and the distance grows if we consider papers with authors in industry; Europe lies in the middle, with the rest of the continents with little or no presence.

Top research institutions. We identified the top 10 research institutions based on the affiliation of the first author of the papers (see Figure 6). It is worth mentioning that 23 authors had two affiliations and, in these cases, each of their affiliations counted as 1/2. Analogously, one author had three affiliations, and each of his affiliations counted as 1/3. As shown, the leading research

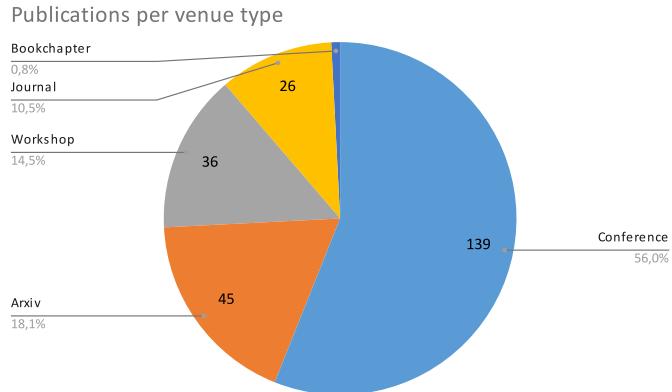


Fig. 3. Number of publications per venue.

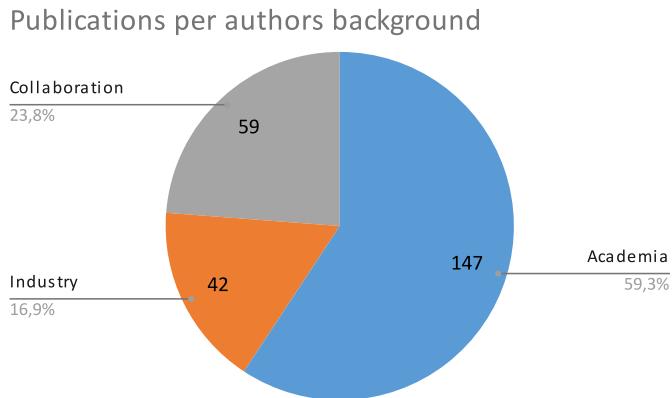


Fig. 4. Number of publications per author affiliation.

institution is IBM, followed by National Institute of Informatics of Tokyo, University of California at Berkeley, Carnegie Mellon University, and Google.

4.2 Characteristics of the Primary Studies

From the 248 studies analyzed, 156 were empirical studies. We classified these empirical studies according to the research method reported. Table 3 provides the list of research methods, the criteria used to classify them, and the number of papers for each type of research method. Interestingly, the most common type of study is those who provided one (or more) case studies (37.8%). We also notice that SE4AI has a significant number of benchmarks (22.4%) which might be explained due to the data-driven nature of the field.

To provide some context to the distribution of papers by research method in SE4AI, we may compare the obtained results with the bibliometric assessment of SE by Wong et al. [216]. As part of a series of bibliometric reports on SE, they analyzed SE papers from 2013 to 2020. It is worth noting that the classification of papers by research method in [216] relies on a mapping of terms found in the titles of the primary studies only. Hence, a formal comparison with our findings in quantitative terms cannot be performed directly. Nevertheless, their results are quite consistent with ours. From 11,500 papers analysed in [216], the most common research methods in SE are (as

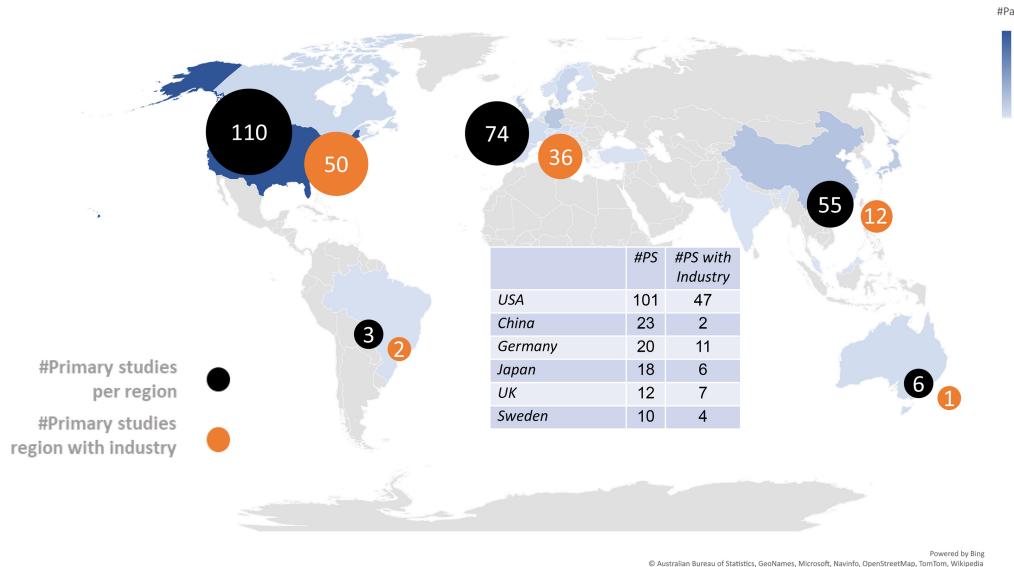


Fig. 5. Distribution of publications by continent and country.

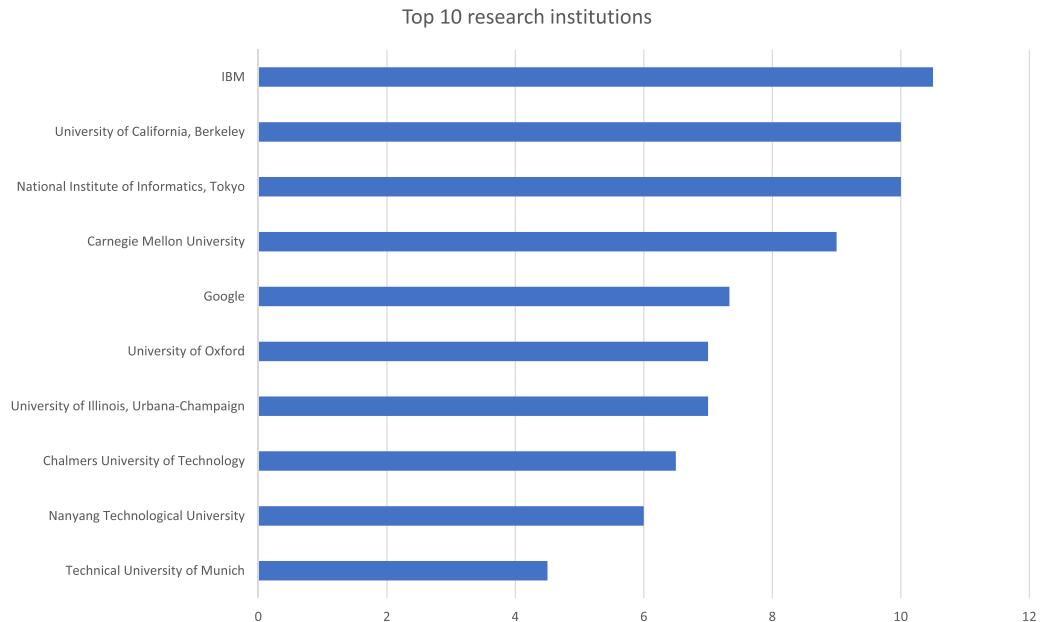


Fig. 6. Number of publications by first author's affiliation (top 10 institutions).

reported in the titles of primary studies): case studies (315 papers), experiments (194), literature reviews (179), and surveys (138), among others (e.g., simulation, theory, systematic mapping).

This distribution of papers by research method is in line with our findings for SE4AI, where the top research methods in SE4AI are: case studies (59 papers), benchmarks (35), experiments (23), and surveys (14). The only differences are that we do not include “literature reviews” (which is part of our Exclusion Criteria - EC4) and that we found a significant number of benchmarks.

Table 3. Number of Papers Per Type of Research Method

Research method	Classification criteria	#papers	%
Case study	The study reports that it employs a case study or an exploratory study where the researchers analyze and answer predefined questions for a single or multiple cases.	59	37.8%
Benchmark	A rigorous study that evaluates or compares one or more algorithms in well established operational settings (with data and variables).	35	22.4%
Experiment	An empirical enquiry that investigates causal relations and processes.	23	14.7%
Survey	The study reports that it employs a survey through a questionnaire, observation, or interview.	14	8.9%
Mixed method	The study reports using several research methods or a mix of research methods.	5	3.2%
Controlled experiment	The study mentions that it employs an empirical enquiry that manipulates one factor or variable of the studied setting.	4	2.5%
Action Research	The study reports employing action research. That is, a research idea is applied in practice and the results are evaluated (a crossing between an experiment and a case study).	1	0.6%
Other/Not stated	Other research methods, or the research method was not stated in the empirical study.	15	9.6%

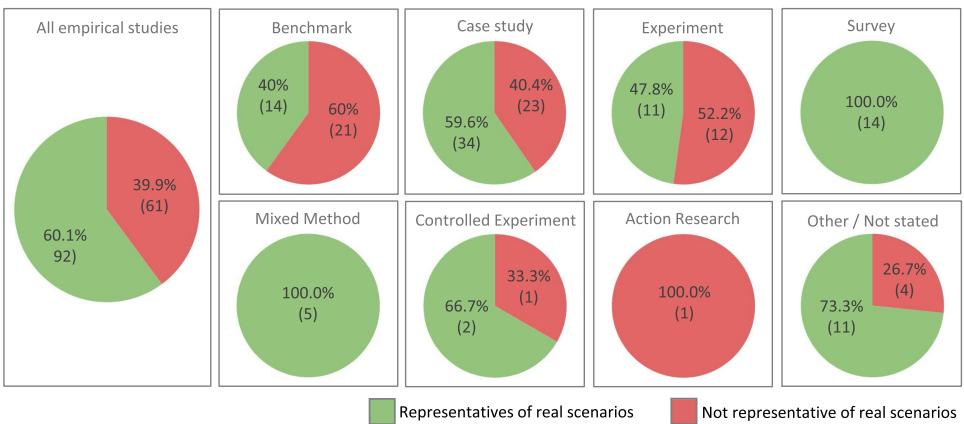


Fig. 7. Subjects/Objects of study in empirical research studies.

However, we must acknowledge that none of the papers classified as benchmarks in our study included the term benchmark in the title.

For the quality assessment, we applied Ivarsson and Gorscheck's rigor and relevance quality assessment model [93]. From the different metrics proposed in this model, we report here only those in which we feel confident that they have been measured in the most objective and consistent manner by all the different authors of this study. In other words, we found that, despite several efforts, evaluating some metrics is prone to subjective interpretation (e.g., measuring if a study design is described with enough rigor) and we did not feel confident that the results of those metrics were consistent enough to provide a reliable analysis. The metrics that we have finally incorporated in the report of the results are: the realism of the study environment, the scale, and whether they included threats to validity.

4.2.1 Evaluation of the Realism of the Study Environment. Figure 7 presents the evaluation of the realism of the study environment in terms of the subject/object used in the empirical study. In 60.1% of empirical research studies, the subjects and/or objects used in the evaluation are representatives

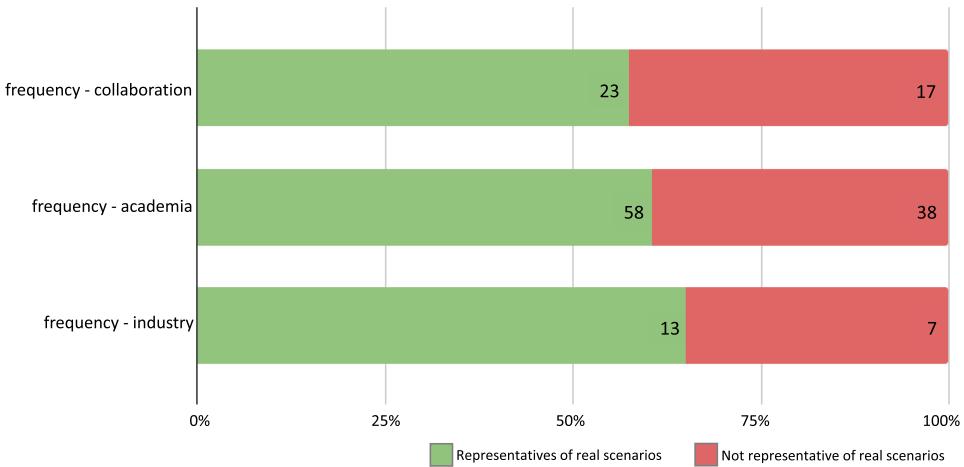


Fig. 8. Subjects/Objects of study in empirical research studies by authors' affiliation.

of industrial professionals and industry systems or real data sets). Conversely, 39.9% of papers used scenarios based in students and/or simulated data settings.

Analyzing these results by type of study, we see that all surveys and mixed methods address representative subjects/objects. Although this is not surprising for surveys, as they generally approach subjects with the required background, it is quite interesting to see that also mixed methods have such a high degree of real subjects/objects (although, given the low number of studies of this type, this is not a firm conclusion and should be taken with care). In contrast, papers using more non-representative subjects/objects are benchmarks (60%) and experiments (52.2%) (we refrain from drawing conclusions on action research, as in this type of study $n = 1$). These results are not surprising considering the nature of this type of studies (especially in the case of experiments).

Considering the authors' affiliations (see Figure 8), we observe that papers written by authors from the industry have a slightly higher percentage of using real scenarios compared to collaboration or purely academic papers.

4.2.2 Scale of the Application Used in the Evaluation. Figure 9 presents the scale of the application used in the evaluation. In 37.3% of the empirical studies, the scale of the application used was of industrial scale. In 43.8% of cases, the evaluation included some reference to the industry (e.g., data from a company) and in 19.0% the scale was of laboratory data (i.e., toy examples).

If we analyze the results by type of study, we observe that surveys and mixed methods present mostly evaluations of realistic size (for mixed methods, this is not a firm conclusion due to the low number of studies of this type, and again, we refrain from drawing conclusions on action research). Benchmarks, case studies, and experiments usually use some data from the industry (68.6%, 42.1%, and 47.8%, respectively) but those with a scale of realistic size are very low (14.3%, 35.1%, and 30.4%). The rest of types of study do not have enough studies to present any valid conclusion.

Considering the authors' affiliation (see Figure 10), we observe that most of the papers written by industry authors used an application of realistic size, whereas papers from both collaboration and purely academic authors had most of the papers including some data from the industry but without applying it into a realistic size environment. Finally, academic papers were the ones that had most of the papers using toy examples.

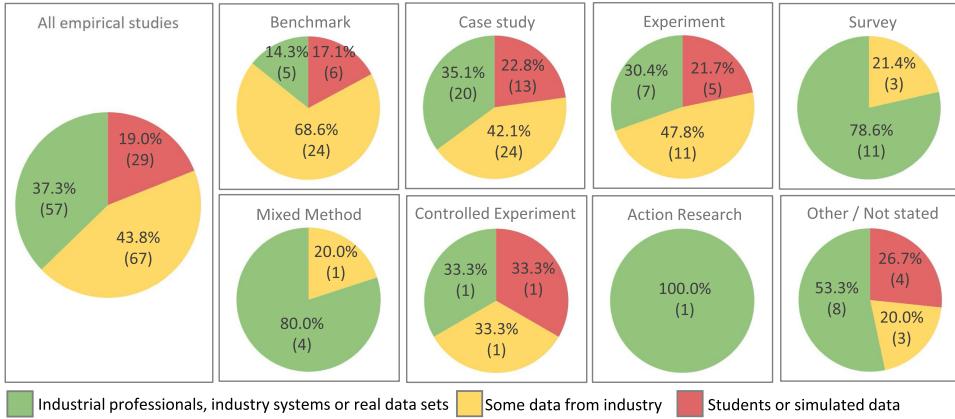


Fig. 9. Scale of the application used in the evaluation.

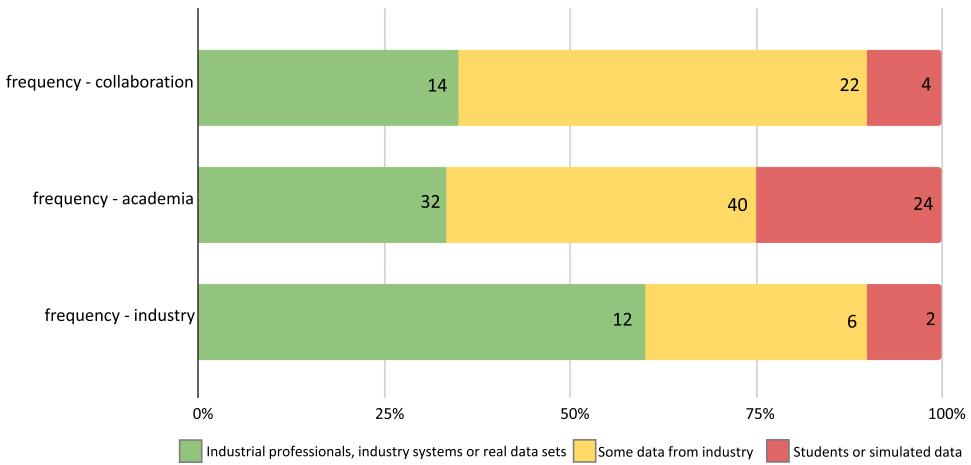


Fig. 10. Scale of the application used in the evaluation by authors' affiliation.

4.2.3 Threats to Validity in the Primary Studies. Strikingly, 65.4% of the empirical research studies do not provide any threat to validity. Only 17.6% of these studies have the validity of the evaluation discussed in detail, and the remaining 17.0% just briefly mention them (see Figure 11).

If we analyze these results by type of empirical study, we notice that research papers presenting case studies, experiments and benchmarks are the ones where threats to validity are mostly ignored. Most of these research papers do not even mention threats to validity, ignoring them in 75.4%, 73.9% and 65.7% of the cases, respectively. In contrast, those discussing more in detail the threats to validity are controlled experiments and mixed methods (again, this conclusion should be taken with care due to the low number of papers of these types).

Analyzing the results by authors' affiliation (see Figure 12), we observe that the vast majority of papers from collaborations and industry do not discuss threats to validity. But surprisingly, most of the academic papers do not discuss threats to validity either, even though the frequency of papers that discuss threats to validity is higher than its counterparts.

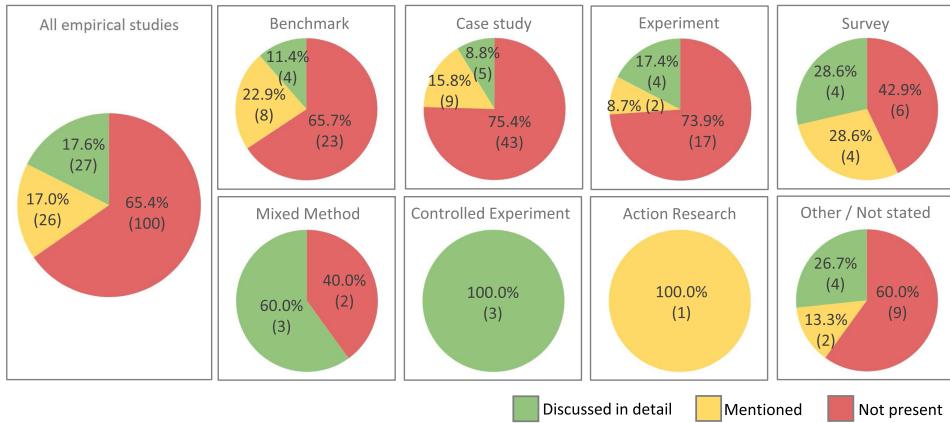


Fig. 11. Threats to validity in empirical research studies.

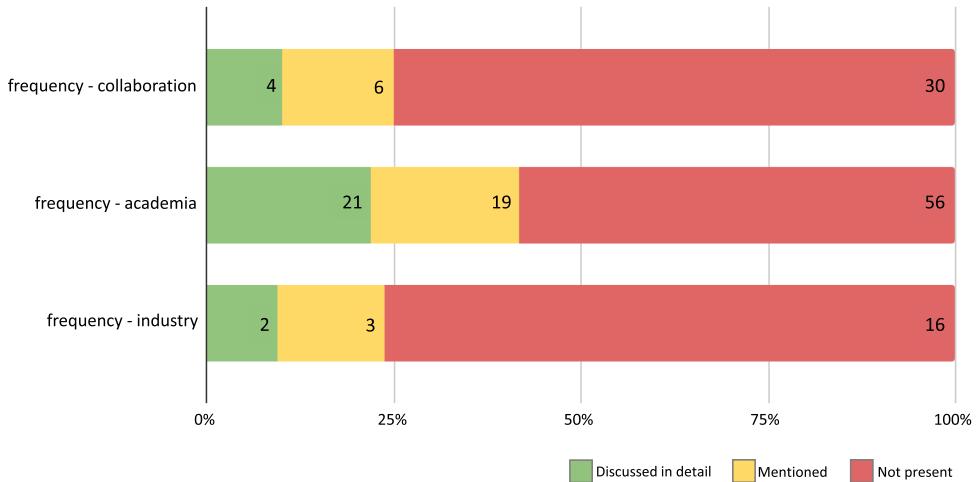


Fig. 12. Threats to validity in empirical research studies by authors' affiliation.

4.3 Discussion

Below, we report the main observations and take-away messages for this RQ:

Observation 1.1: SE4AI is an emerging research area. Not only the growing annual publication trend supports this observation, but also the distribution in type of venues, with the importance of arXiv and the low percentage of publications in the form of journal papers (26 papers, i.e. 10.5%; with 23 of them published in the last three years). We may expect that a number of these arXiv publications will become archival publications in journals, contributing to the gradual consolidation of the field.

Observation 1.2: Literature reviews in SE4AI need to consider arXiv. As a follow-up of the previous observation, we may affirm that literature reviews in the SE4AI area cannot be limited to automatic searches in typical digital libraries (Scopus, Web of Science, or publisher digital libraries as ACM DL or IEEE Xplore) because arXiv papers will not be found. Either arXiv papers are manually added to the results, or snowballing is used, as we have done in our study.

Observation 1.3: Research in SE4AI involves more industry authors than usual. Our study has identified 101 industry and collaboration papers, representing 40.7% over the total. We have compared these numbers with other recent literature reviews on topics that can be considered of practical importance, in which we find lower percentages, e.g., 25.6% in Management of quality requirements in agile and rapid software development [19] or 28.7% in open-source software ecosystems [61]. This observation is important to argue for practical applicability of the findings reported in the primary studies found in our literature review. Major players in the industry are big companies like Microsoft, IBM or Google.

Observation 1.4: Industry involvement is especially significant in Europe and North America. Looking in more detail the results of industry involvement, the percentage grows significantly in North America (45.5%) and Europe (48.6%) compared to Asia (21.8%). This difference becomes more apparent if we compare the two countries with the highest numbers of studies, the USA (46.5%) and China (8.7%), showing two different approaches to research.

Observation 1.5: Industry involvement slightly improves the realism of case studies, and significantly improves its scale. Industry papers have just a slightly higher percentage of realistic scenarios compared to collaboration or academic papers. In contrast, we observe that the authors' background affects more significantly the scale of the evaluation. In this regard, industry papers use bigger scales in the evaluations, with approximately, twice as much as collaboration or academic papers. Conversely, academic papers use more toy examples compared to its counterparts, with approximately three times as much as collaboration or industry papers.

Observation 1.6: Threats to validity are mostly ignored, even for papers from academic authors. As observed, most empirical studies do not discuss threats to validity. Papers from academic authors have a higher frequency of papers discussing threats to validity compared to collaborations or industry papers, but they are still a minority, and most of the academic papers ignore threats to validity, which compromise the quality of the research. This may be caused by the number of arXiv and workshop papers, which tend to discuss fewer threats to validity than journal or conference papers.

5 RQ2: WHAT ARE THE CHARACTERISTICS OF AI-BASED SYSTEMS?

This section respectively discusses the terminology used on the primary studies as well as the dimensions in which we have classified them, and the key quality attribute goals of AI-based systems.

5.1 What is an AI-based System?

We found a large variety of terms used in the primary studies. In Table 4, there is an overview of all terms used more than twice to discuss the type of system investigated in the corresponding primary study. We observed a mix of very general terms (such as "machine learning" or "AI technologies"), specific AI technologies (such as "deep neural networks" or "ML libraries"), and AI application domains (such as "robotics system" or "automotive system"). We furthermore noticed that AI seems to be always used in terms of learning components and not including rule-based expert systems. This corresponds to the new wave of AI associated with learning from data.

In the further inductive coding of the terms and study objects, we distilled three dimensions that can be used to classify the contributions of primary studies about AI-based systems.

The first dimension is the **Scope** of the system under analysis. In particular, the scope refers to the question: *How is AI implemented inside the system?* AI can either be one component in a system ("component"), dominating the entire system ("system"), implement one or more particular algorithms ("algorithm", such as DNN), or providing a pipeline or infrastructure ("infrastructure", e.g., PyTorch).

Table 4. Terms Used in the Primary Studies to Refer to AI-based Systems with “Intelligent” Components

Count	Terms (comma-separated)
43	Machine learning
40	ML system
28	Deep neural networks
27	ML algorithms
23	ML models
21	Autonomous vehicle
19	Autonomous systems, ML components
18	AI systems
17	AI, Neural network
16	Deep learning systems, ML application
15	ML techniques
14	Autonomous driving system
11	Cyber-physical systems with Machine Learning components (CPSML)
10	ML software, ML-based system
9	AI-based system
8	DNN-based software
7	Deep learning, Reinforcement Learning (RL)
6	AI software, Machine learning classifiers, ML program
5	Artificial Neural Networks, Classifier, Intelligent systems
4	AI components, AI model, DNN model, ML methods, ML pipeline
3	AI applications

The second dimension is the **Application Domain**. Many studies do not concern themselves with what the AI will be used for exactly but study, for example, DNNs in general. There are, however, also several studies focusing on concrete applications of AI. The most frequent of those is the domain of autonomous vehicles that encompasses a part of the autonomous systems and the autonomous/automated driving systems from Table 4. An application domain can also be more generic, such as ML frameworks.

The third dimension is the **Technologies of AI** under consideration. AI technologies can be, for example, ML in general or DL methods. This is usually stated in some way, and we also think it is important because it can make a huge difference in how generalizable the results of a primary study are. For instance, test approaches for AI components that use random forests might not be useful for AI components built on DNNs.

We then used this structure to code all our primary studies (not only those that provided definitions) to get insights into what exactly they investigated. We found that almost half of the primary studies look at SE4AI at the system level (see Figure 13). This means that they investigate complete systems such as autonomous cars regarding their AI aspects. A quarter of the primary studies focuses on the AI components directly. An example would be the image recognition component in an autonomous car. The remaining primary studies either investigate or propose methods for specific algorithms (such as DL) or for AI infrastructure (such as TensorFlow) without considering specific applications.

In Figure 14, we show the number of publications in different domains structured by the AI technology investigated. The only dominant application domain is automotive with its hype on

Dimension 1: Scope

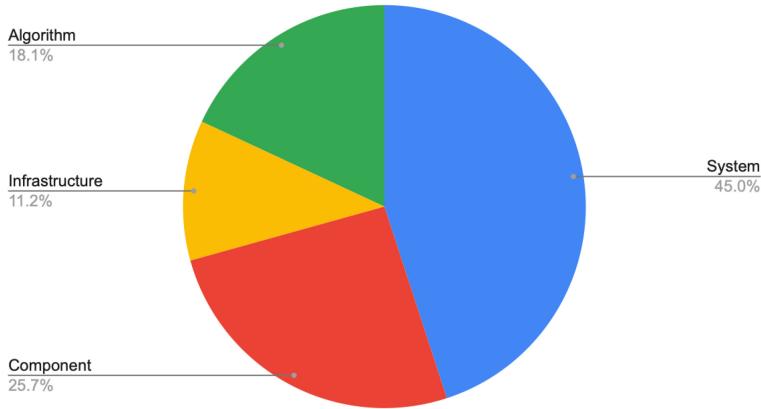


Fig. 13. Scope of the research in the primary studies.

autonomous cars. More than a quarter of the primary studies aim at this domain. Example systems are the pedestrian detection system in autonomous driving at Bosch [66] or an automated emergency braking system at IEE S.A. (Luxembourg) [1]. Almost half of the primary studies look at AI in a generic way without considering any application domain. The further many domains that we found only make up between 0.8% and 3.3% of the primary studies. Some domains also overlap, as there are embedded systems in automotive or aviation. Examples for these further domains include WeChat's NMT system for automatic machine translations [232], the pin recommender system at Pinterest [123] or the CognIA chatbot for financial advice from IBM [204].

5.2 What are the Key Quality Attribute Goals for AI-based Systems?

Many of the primary studies focused on one or several software product or process qualities, e.g., by proposing an approach that improves a certain quality attribute or by analyzing the context of a certain quality for AI-based systems. We therefore extracted the quality goals per study ($0 \dots n$) and could assign at least one goal to 190 out of 248 studies. These study goals were then harmonized for consistent terminology, until we ended up with 40 different terms (see Figure 15 for the most frequent ones). In total, these terms were mentioned 378 times, i.e., each study was linked to an average of 1.5 quality attribute goals. We then analyzed this mapping to identify trends and generalizations. For the analysis, we first identified the level of abstraction per goal (vertical analysis) and then formed thematic clusters of semantically related goals (horizontal analysis).

Regarding the vertical analysis, the identified publications discuss 40 quality attributes at various *levels of abstraction*, which we classified with the three labels low (very specialized low-level attributes, e.g., sub-QAs in ISO 25010), medium (top-level attributes in ISO 25010 or similar granularity), and high (very abstract QAs or aspects encompassing a broader range of qualities). Exactly half of them (20) were classified as low, 13 as medium, and 7 as high. This indicates that the majority of approaches aims to improve quality attributes at the abstraction level specified in ISO 25010 or the level below, with hardly any studies targeting more abstract or broad-range qualities. The most frequently discussed quality attributes include *functional correctness* (40 mentions) at the lowest level, *safety* (74 mentions) at the medium level, and *trust* (8 mentions) at the highest level.

For the horizontal analysis via thematic clustering, we identified a total of nine clusters (see Figure 16). Except for the goal for general quality attributes, each of the 40 goals and their associated studies are assigned to exactly one of these clusters. By far the most prominent cluster is

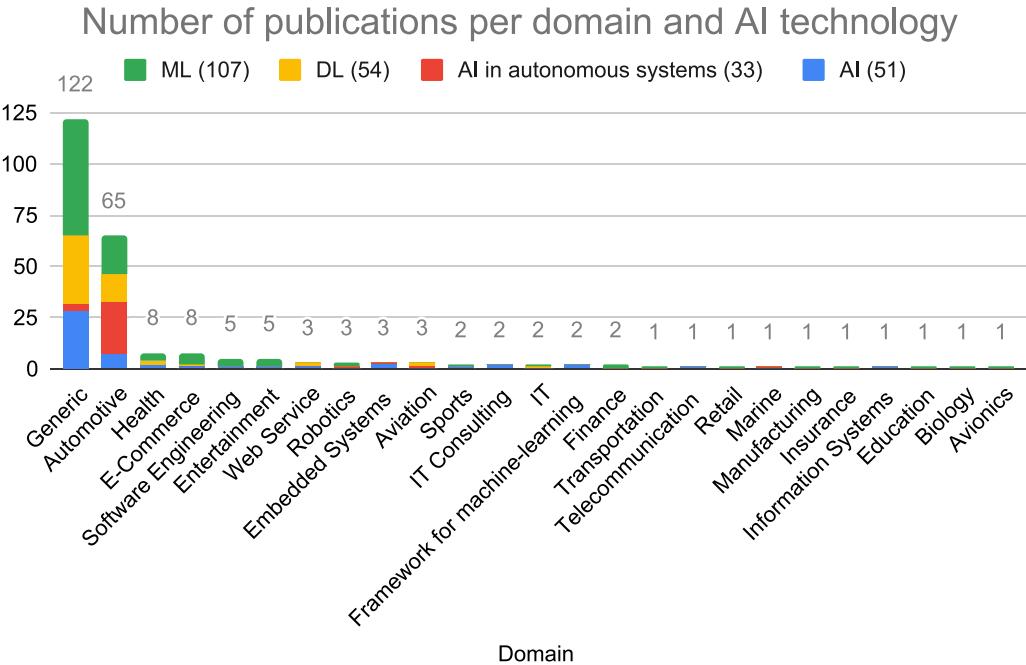


Fig. 14. Domain and AI technology investigated in the primary studies.

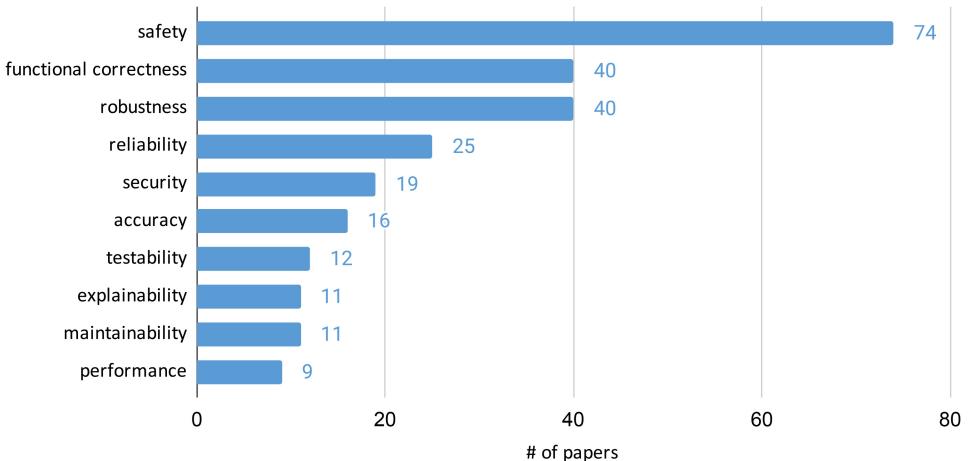


Fig. 15. Most frequent quality goals of AI-based software systems.

dependability & safety, which accounts for a total of 179 mentions. It includes four of the five most frequently mentioned individual QA goals, namely *safety* (74), *robustness* (40), *reliability* (25), and *security* (19). With 68 mentions, *functional suitability & accuracy* is the second-largest cluster, followed by *maintainability & evolvability* with 38 mentions. The remaining six clusters are smaller (7–25 mentions). Overall, we identified a strong focus on qualities related to safety (especially in

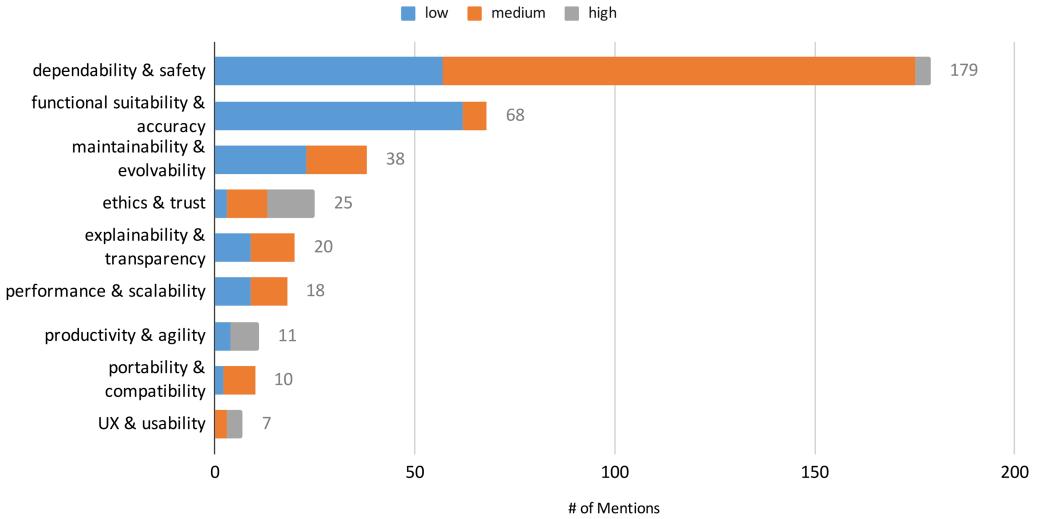


Fig. 16. Frequency of quality goals per thematic cluster and level of abstraction.

the context of smart cyber-physical systems like autonomous vehicles) and correctness & accuracy, which was analogous to the many studies on AI software testing.

5.3 Discussion

Observation 2.1: AI is commonly associated with DL and considered as part of a complex software system. We propose that any paper that fits into our inclusion criteria discussing the application of SE to AI-based systems should make explicit what exactly they consider on our identified dimensions. This would be a first step in making the scope and limitations of primary studies more clear.

Observation 2.2: Many various synonymous terms are used to denote a system or component that uses some kind of AI or ML. Commonly used synonyms used to refer to software systems, which use AI technologies include AI-based system, AI-enabled system, AI-infused system, AI software, ML solution, or DL system. We propose to use the following definitions to guide the selection of terms for future studies:

- **AI component:** A part of a system. The component uses AI to some extent. Examples range from a component whose behaviour depends to some extent on some embedded AI code, or an AI library as a special AI component that provides a concrete implementation of AI algorithms.
- **AI-based system:** A system consisting of various software and potentially other components, out of which at least one is an AI component.

Observation 2.3: There exists diversity in the terminology to refer to AI-based systems, making unclear what is the object of the research. We propose that SE4AI papers should use a taxonomy that makes clear what kind of AI-based system is investigated and how general the methods and approaches are supposed to be. We use as an example the paper by Burton et al. [32]. They use the term “machine learning” in the title, but then focus on Convolutional Neural Networks. We depicted a corresponding taxonomy in Figure 17. On the top level, we keep as close as possible to established discussions of the terms. For that, ML is commonly seen as a part of AI

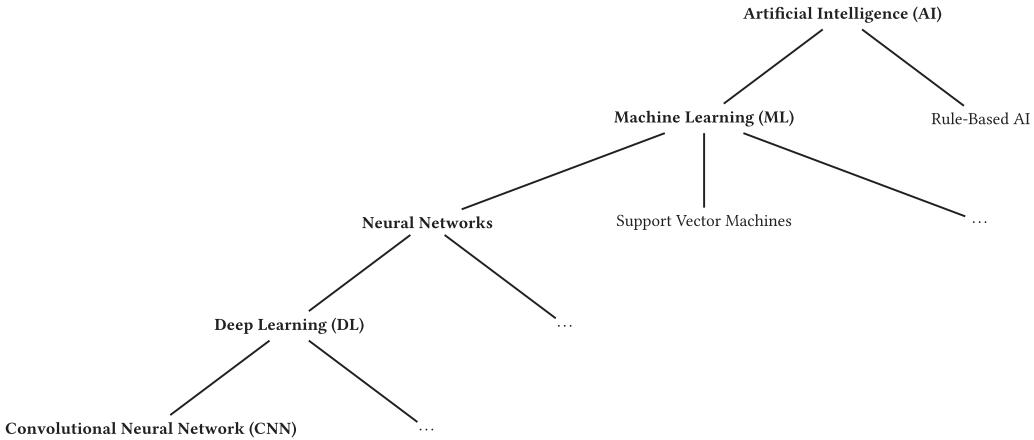


Fig. 17. Example taxonomic classification for paper [32].

([111]). Hence, an ML-based system is also an AI-based system. Using the DARPA terminology (<https://www.darpa.mil/news-events/2018-07-20a>), in the first wave of AI, there were mostly rule-based systems. The second wave added statistical learning. We see only papers about AI in at least the second wave sense in our primary studies. Similarly, on the next level, in ML most primary studies investigate neural networks, mostly DL. So, also for Burton et al., we would go further and also refine it to the next level “Convolutional Neural Network” as a specific type of DL. Using such a taxonomic classification and explicitly mapping contributions to their respective levels would make the article clearer. We suggest that such a taxonomic classification would be useful for all SE4AI papers.

Observation 2.4: Most of the terms used are not defined explicitly. Most commonly defined terms include “deep learning system” (5), “ML system” (5), “AI” (4), and “machine learning” (4). Besides using a taxonomy, we propose explicitly defining the terms used.

Observation 2.5: Most primary studies focus on software systems. In the analyzed studies, AI is not just one of its many components of a software system, but typically constitutes its dominating part.

Observation 2.6: The most studied properties of AI-based systems are dependability and safety. Overall, we identified a strong focus on qualities related to safety (especially in the context of smart cyber-physical systems like autonomous vehicles) and correctness & accuracy. However, there are research gaps for less studied properties, such as usability, portability, or particularly important in the context of trustworthiness and the understandability aspect. More importantly, inherent and critical quality characteristics in AI-based systems such as explainability and transparency require the attention of researchers to assure the high maturity level required by industry.

Observation 2.7: The use of ML and DL has only been extensively used in AI-based systems of the automotive domain, and to a much lesser extent in healthcare and e-commerce. We believe that the increasing and successful application of ML and DL (e.g., image-, language- classification, and object recognition) in the automotive domain shall inspire researchers and practitioners to explore and apply it in other domains. Based on the primary studies, many domains have received little or no attention (see Figure 16).



Fig. 18. The 248 primary studies classified into 11 SWEBOK Knowledge Areas based on their SE contributions for AI-based systems.

6 RQ3: WHICH SE APPROACHES FOR AI-BASED SYSTEMS HAVE BEEN REPORTED IN THE SCIENTIFIC LITERATURE?

We classified the 248 primary studies in 11 SWEBOK areas (see Figure 18). Primary studies were thematically associated with at least one SWEBOK area based on their research directions and contributions. For eight areas, we derived subcategories to organize the research further. We did not do this for areas in which there were two or less primary studies. In the following subsections, we provide a detailed overview of SE approaches for AI-based systems in each SWEBOK area and illustrate them with exemplary papers.

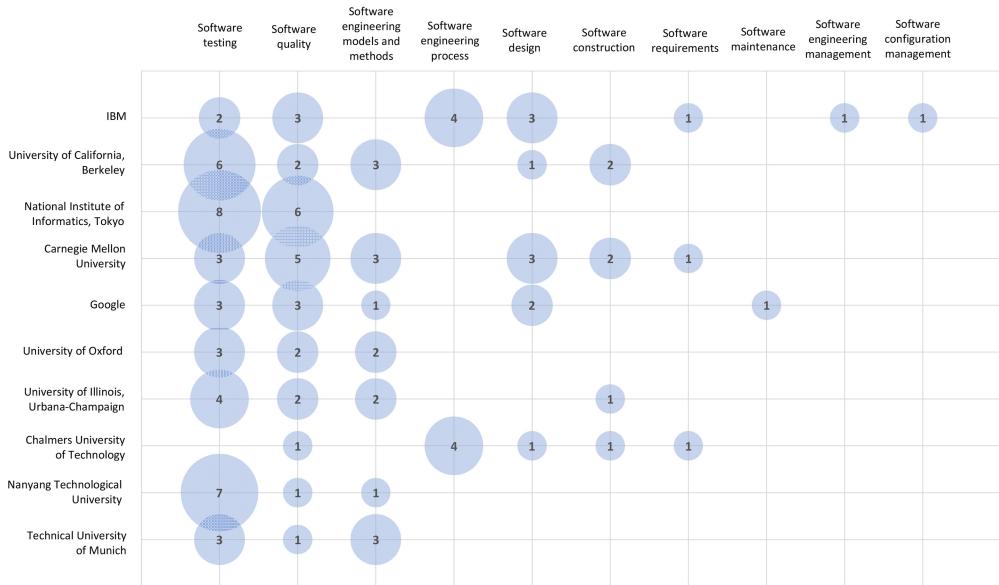


Fig. 19. SWEBOOK areas addressed by the leading institutions in SE4AI.

We also analysed the SWEBOOK areas addressed by the leading research institutions in SE4AI (see Figure 19). In terms of research topic, we find a two-fold situation. Three institutions (University of California, Berkeley; National Institute of Informatics, Tokyo; and Nanyang Technological University) are focused on one particular SWEBOOK Knowledge Area, namely Software Testing, while Chalmers University of Technology conducts research mainly in the Software Engineering Process area. The rest of institutions cover a wider variety of topics, and particularly IBM and Carnegie Mellon University have published research related to seven and six SWEBOOK Knowledge Areas, respectively.

6.1 Software Requirements (17 Studies)

For *software requirements*, we identified 17 studies. Based on our thematic analysis, we derived five subcategories, with most papers belonging to several categories.

Nine papers address the Requirements Engineering (RE) *process* for AI-based systems. Vogelsang and Borg are the only ones to cover the complete process, from elicitation to verification and validation [205]. They summarized important characteristics of the RE process for ML systems, such as detecting data anomalies or algorithmic discrimination. The rest of the primary studies concentrate on one particular RE activity, with *specification* being the most popular one (six papers). Four of these papers focus on non-functional requirements (NFRs, see below for details). Further examples include methodological aids such as the notion of supplier's declaration of conformity [12] or conceptual frameworks to improve the specification of requirements for explainable [186] or safe [15] AI-based systems. The two general papers on specification focus on formal aspects related to ambiguity [161] and the need to consider partial specifications for AI-based systems [171]. Two other papers addressed requirements-driven *derivation*, in both cases in automotive systems, but with different aims: while Burton et al. derive low-level requirements from safety goals [32], Tuncali et al. derive test cases from safety and performance requirements [199]. The

last paper focuses on the *elicitation* of safety requirements in automotive systems using a risk-based approach [2].

Four papers in the RE process category also propose a particular *notation* to express requirements. The aforementioned two papers on derivation defined a concrete notation to make the derivation process less ambiguous, namely by using goals in the case of Burton et al. [32] and temporal logic in Tuncali et al.'s approach [199]. Adedjouma et al. also employ goals to represent safety risks and hazards [2]. The fourth paper expresses requirements as linear arithmetic constraints over real-valued variables to support satisfiability modulo theories (SMT) [121].

Eight papers did not provide concrete RE approaches, but were more *conceptual* in nature: the authors tried to provide a foundation for AI software RE research by disseminating current *practices* and/or *challenges*, e.g., via the use of interviews [205] or surveys [228]. Many of these papers analyzed how RE for AI changed in comparison to traditional systems, and especially what issues currently prevent effective practices [20, 82, 110]. In addition to that, Otero and Peter also tried to provide research directions to address some of these challenges [152].

Finally, three very diverse papers discuss *RE foundations* for AI-based systems. As mentioned above, Salay and Czarnecki introduced foundations for partial specifications as appropriate for specifying AI-based systems [171]. Meanwhile, Otero and Peter propose a model of requirements for Big Data analytics software [152]. Lastly, Arnold et al. incorporate traceability into their requirements specification approach [12].

As an additional facet to the above subcategories, 10 of the 17 studies focused specifically on *NFRs*. While both Horkoff [82] and Kuwajima et al. [118] targeted NFRs in general, the other papers were concerned with one or a few specific NFRs, e.g., safety [2, 15, 121, 171] or model performance [12, 32, 199]. Arnold et al. also included security requirements in their FactSheets approach [12], whereas Sheh and Monteath were the only ones to study the nature of requirements for explainable AI [186].

Key takeaways for software requirements:

- A lot of focus on NFRs for AI (10 of 17 studies), especially new AI-specific quality attributes
- Several specification and notation approaches to deal with probabilistic results or ambiguity, e.g., partial specification
- Very few holistic views on the RE process for AI (only one study), focus on support for RE specification and derivation

6.2 Software Design (34 Studies)

In the area of *software design*, we formed seven categories to group the 34 studies. One of the largest of these – *design for X* – is concerned with design approaches or techniques to improve or assure one specific quality attribute in AI-based systems. It comprises 11 papers, of which the majority is concerned with *safety* (seven papers). Most of these studies propose design strategies for AI systems in safety-critical domains where unsafe behaviour can have immense negative consequences, like the use of architectural components as a protection layer for safety in autonomous vehicles [139], design best practices for safety verification methods [16] or behavior-bounded assurance [174] for autonomous aerial vehicles, safety design strategies (e.g., inherently safe design, safety reserves, safe fail, and procedural safeguards) for ML components [118, 202], and specific safety strategies for cyber-physical systems [7, 203]. The remaining four design approaches are for

reliability [130], *user experience* [222], and for quality attributes related to ethical AI like *fairness* [12, 114].

Another large category is *frameworks or platforms* (11 studies), ranging from a generic level of abstraction for end-to-end ML application development and deployment [14, 40, 85, 100, 114] to available tool support for managing reliable ML applications [18], packaging and sharing ML models as reusable microservices [230], and the development and deployment of ML applications [65]. Several platforms follow model-driven engineering principles [76], for instance a model representation for production ML models [45]. These types of platforms are relevant for the cyber-physical systems domain, for which we also found dedicated tool support [76, 167].

With six papers, the third-largest category is formed by studies that elicited and described *design challenges* for AI-based systems, e.g., by conducting empirical studies like surveys [228] or reporting industry experiences [75]. The scope of the described design challenges varies greatly and covers areas such as intelligent automotive systems [120], ML model management [177], or ML fairness [81]. The details of these challenges are explained in Section 7 (RQ4).

We also found four papers on *design patterns* for AI-based systems: an architecture pattern to improve the operational stability of ML systems [225], patterns to improve the safety of systems with ML components [182], an architecture pattern to manage N-version ML models in safety critical systems [129], and finally, more abstract solution patterns to address recurrent business analytics problems with ML [146].

Finally, we found two studies about technical debt and the design stage, for which we created the category *design debt* [5, 60], and resources for teaching software design for AI-based systems [119].

Key takeaways for software design:

- Many design strategies to cope with specific quality attributes, e.g., with safety or reliability
- Several concrete AI infrastructure proposals, e.g., for sharing models as microservices
- However, at the system level, there are few proposals for patterns, design standards, or reference architectures

6.3 Software Construction (23 Studies)

A total of 23 studies were concerned with *software construction*. Many of these studies either provided specialized *tools* (six papers) or more holistic *platforms* (four papers) to support the development of AI systems. Exemplary application contexts for tools were the deployment and serving of general prediction systems [42], lowering the barrier for using ML techniques [14, 99, 100, 155], or model-based development toolchains [76]. For platforms, the common goal was to provide a more comprehensive infrastructure to improve and accelerate the AI development workflow [14, 40, 178], sometimes in more specialized domains like safety-critical robotics [47].

Furthermore, eight studies reported construction *practices and guidelines* for AI systems based on diverse experiences, such as implementing ML components to detect and correct transaction errors in SAP [162], a systematic comparison of DL frameworks and platforms [73], experiences from improving Airbnb search results with DL [75], experiences from 150 ML applications at Booking.com [21], AI model criteria relevant for end users [58], automatic version control in notebooks [99], or practices collected via practitioner surveys and/or interviews [206, 228].

Similarly, seven studies reported current *challenges* in constructing AI-based systems, most of them focusing on DL. Challenges have been derived via StackOverflow questions [92, 227], surveys [228], theoretical analyses of the AI development process [120, 218], or case studies [13, 142].

Lastly, one paper was concerned with the software construction of AI systems in a *teaching* and education context [119]: Kästner and Kang describe their “SE for AI-enabled systems” course material and infrastructure and share lessons learned from educating Master students in this field.

Key takeaways for software construction:

- Many state of practice studies synthesized construction challenges and guidelines to address them.
- Several tools and platforms have been proposed to improve AI development activities, but their maturity, rationales for their selection, and level of adoption remain vague.

6.4 Software Testing (115 Studies)

We identified 115 studies focused on *software testing*. During the analysis, we formed 11 subcategories (*bugs & debugging*, *challenges*, *explanations*, *quality testing*, *test architecture & languages*, *test case*, *test case generation*, *test case selection*, *testing methods*, *testing metrics*, and *generic*), which we partially refined into sub-themes. For each of the 115 studies, we assigned one or more of these subcategories. In three cases, we assigned five different subcategories to a paper. The different subcategories are described below:

Four papers addressed *bugs & debugging*. Three of these papers conducted empirical studies examining bugs in ML projects [91, 190, 229], whereas one paper proposed a specialized debugger for ML models [34].

A total of 12 papers studied the *challenges* in software testing for AI. Nine of them discussed the challenges, issues, and needs in AI software testing based on the current state of the art, either for generic AI systems [28, 64, 83, 152, 177] or focusing on the particular challenges for autonomous vehicles or other safety-critical systems [106, 118, 120, 172]. Finally, three proposals identified the challenges for generic AI or ML systems through empirical methods like questionnaire surveys with practitioners [81, 90, 228].

Two papers addressed testing-related *explanations* for ML systems [113, 150]. Due to the difficulty to understand the results of ML systems in some scenarios, these papers provided a method for explaining how the ML system reached a particular result, including failures or how the tester addressed them to correct the ML system.

In 35 papers, the focus of the presented testing approach was aimed at improving very specific quality characteristics of the AI-based system under test (subcategory *quality testing*), including safety (e.g., [10, 37, 170]), robustness (e.g., [55, 98, 189]), security (e.g., [26, 51, 174]), fairness [3, 81, 200] or others (e.g., [36, 108, 168]). Safety was indeed the most addressed quality characteristic with 21 proposals, followed by robustness and security with seven and four papers, respectively.

In the subcategory *test architecture & languages*, we identified one paper proposing a new testing architecture [149], and one paper proposing a specific testing language [131].

17 papers were assigned to the subcategory *test case*. The type of test cases that these studies addressed were adversarial test cases in 14 occasions (e.g., [49, 55, 72]) and corner test cases in 3 occasions [22, 217, 223]. As opposed to the related categories about test case generation and

selection, papers in this general category were very focused on conceptualizing different types of test cases.

Regarding *test case generation*, 40 papers provided automatic means for the generation of test cases. Most of them augment existing test cases, deriving new tests from an original dataset (e.g., [48, 97, 196]). Some of these proposals generate these test cases randomly (e.g., [131, 231]), but others focus on attaining specific objectives when generating test cases, like generating corner case testing inputs (e.g., [223]), adversarial testing inputs (e.g., [55, 221, 233]) or increase the coverage of the test suites (e.g., [50, 127]). Other approaches generate test cases with discriminatory inputs to uncover fairness violations [200], or with specific inputs to uncover disagreements between variants of an AI/ML model [220]. Approaches like [215] generate test suites avoiding too similar test cases to minimize the number of tests to execute. It is worth mentioning that some proposals are based on simulation-based test generation, for instance, to generate tests for autonomous vehicles in simulated environments (e.g., [63, 98, 198]).

A total of 12 proposals were categorized with *test case selection*. Some approaches proposed test case selection techniques as a complementary activity to the test case generation (e.g., [51, 62, 217]). One approach proposed a technique to select test cases based on a metric of importance [67], whereas others proposed techniques to identify corner cases [22], adversarial examples [207] or likely failure scenarios [108]. Finally, a few approaches proposed techniques for test input prioritization to select the most important ones and reduce the cost of labeling [33, 56] or reduce the performance cost of training and testing huge amounts of data [188].

A total of 36 papers addressed *testing methods* for AI systems, following different techniques such as combinatorial testing [105, 127, 131], concolic testing [191, 193, 194], fuzzing (e.g., [48, 151, 219]), metamorphic testing (e.g., [52, 144, 227]), or others (e.g., [35, 49, 128]). From the different methods used, it is interesting to point out that the most popular one is metamorphic testing with 16 studies, followed by fuzzing and mutation testing with six and five studies, respectively.

Moreover, 20 papers focused on the definition and/or exploration of *testing metrics* to measure the testing quality. 14 out of 20 focused on test coverage metrics (e.g., [22, 77, 192]), whereas the rest of the metrics were reported only by one study each: diversity [189], importance [67], suspiciousness [54], probability of sufficiency [35], and disagreement [220].

Finally, 14 papers were categorized as *generic*, as they did not address or contribute to a specific testing theme.

Key takeaways for software testing:

- The main focus in software testing for AI is test cases (55 unique studies), including the two specialized areas test case generation (40) and test case selection (12).
- The majority of papers related to testing methods use metamorphic testing (16 out of 36), followed by fuzzing (6) and mutation testing (5).
- The majority of papers related to testing metrics propose novel coverage criteria (14 out of 20).

6.5 Software Maintenance (6 Studies)

The small *software maintenance* area only comprises six studies, which we group further into three categories. Two studies empirically analyze the nature and prediction impact of *bugs* in AI software [122, 190]. Similarly, two studies are concerned with providing specialized approaches or tool support for the *debugging* of ML software by focusing on explanatory debugging in interactive ML [113], and proposing their Tensorflow debugger based on dataflow graphs [34]. The

remaining two papers elicited and reported maintenance *challenges*, namely [177] as an Amazon experience report in the area of ML model management, and [228] via a questionnaire survey with DL practitioners.

Key takeaways for software maintenance:

- Hardly any studies on the topic. More research is needed, as there are a few open challenges (see next section)
- In addition to state of practice analyses, the focus was on bugs in and debugging of AI-based systems.

6.6 Software Engineering Process (31 Studies)

Many of the studies mapped to the *SE process* area deal with an *AI/ML process* (13 studies). Out of those, a few discuss processes in specific application areas, such as recommendation systems [123]. The remaining ones address processes in general. Although the majority of papers focuses on processes for developing AI-based systems, some papers (e.g., [187]) address the topic of processes for creating new ML algorithms and tools. Four papers [24, 80, 123, 173] present an overview of current practices and challenges faced during AI system development in comparison to traditional software development. For example, Hill et al. [80] conclude from their interviews with developers of AI systems that they generally struggle to establish a repeatable AI development process. Other authors address identified challenges by proposing concrete solutions or a general research agenda for AI systems engineering [24]. Specific approaches include the application of agile development principles to AI model and system development [178], the integration of development and runtime methods known from DevOps [9, 133] or the adaptation of acknowledged process standards such as ISO 26262 [170].

Closely related to the *AI/ML process* is the *AI/ML pipeline* category (6 studies), which instead of the overall AI system development process addresses only the part devoted to creating AI models [6, 38, 44, 65, 85, 201].

We identified three papers that propose *frameworks* for end-to-end support of AI system development [40, 140, 163]. The proposed frameworks target different concepts, e.g., software development, ML, algorithms, and data. They also have been designed for different domains and contexts, such as ML-based health systems [140], accountability improvement via algorithmic auditing [163], and the support of ML solution development within digitalization processes [40].

Furthermore, we identified six studies which provide *tools* to support the SE process of AI-based systems. An example are Patel's general-purpose tools to provide AI/ML developers with structure for common processes and pipelines [155]. More specific use cases are covered by an ML platform to support iterative and rapid development of ML models [178], the DEEP platform with a set of cloud-based services for the development and deployment of ML applications [65], and a toolbox to support data-driven engineering of neural networks for safety-critical domains [38]. Lastly, other works envisioned how these platforms should be implemented [14, 156].

There are also three studies which report *best practices* to build ML-based systems. Mattos et al. propose five tactics to address challenges during the development of ML-based systems [135]: minimum viable and explainable model; randomization; disabling imputation in early stages; automation after the prototype validation; and continuous experimentation. Additionally, experiences on large scale real-world ML-based systems from Microsoft [6] and IBM [4] have led to the proposal of *maturity models*.

As in other SWEBOK areas, several studies talk about *challenges* of AI development processes [59, 102, 123, 125, 135, 138, 173] or a *roadmap* to address them [14]. The challenges are reported later in Section 7.

Key takeaways for SE process:

- Diverse researchers, including R&D from large companies, have investigated the process to develop and maintain AI-based systems. Many of them highlight the need to form multidisciplinary teams for effective AI processes, e.g., including software engineers and data scientists.
- Many analyzed processes have been constructed in an ad-hoc manner based on the early experiences of large companies in AI-based systems.
- However, six studies have focused on AI pipelines at the model rather than the system level.
- Process-related support is emerging with tools (6 studies) and frameworks (3 studies).

6.7 Software Engineering Models and Methods (38 Studies)

From the 38 unique primary studies in this category, the majority formulate concrete proposals on models and methods, while a few also elaborate on challenges, practices, and roadmaps. In terms of topics, the papers lean a little more towards **verification & validation (V&V)** methods (24 papers) than models (12 papers, one of them also in the former V&V methods category), with three papers reporting generic challenges, practices, and roadmaps. In general, there was a strong dominance of safety as a non-functional aspect and application domains like autonomous vehicles in this SWEBOK area.

Most of the papers in the *V&V methods* category (18 out of 24) focus on formally verifying safety of cyber-physical systems, such as autonomous vehicles or robotic systems containing AI components. Typically, the AI components are controlled by artificial neural networks (16 papers), in particular deep networks. Two papers [86, 183] present an overview of current challenges and practices in V&V of autonomous systems, in particular those based on DL. The remaining papers propose alternative V&V approaches to ensure correct, robust, and safe AI-based systems. Part of them aim at systems based on specific types of artificial networks, such as multi-layer feed-forward perceptron [160] or recurrent neural networks [50]; the remaining papers provide generic approaches independent of specific ML methods being evaluated. One group of V&V solutions aim at specific problems of networks being easily fooled by adversarial perturbations, i.e., minimal changes to correctly classified inputs, that cause the network to misclassify them [84]. These approaches explore space of adversarial counter-examples to identify and ensure safe regions of the input space, within which the network is robust against adversarial perturbations [72]. Solutions propose direct search for counter-examples or investigation of input space margins and corner cases. Counter-example or guaranteed ranges of inputs on which artificial neural networks correctly are searched using various optimization techniques. Novelty of these approaches lies often in how the optimization problem is formulated and solved. Computation complexity and scalability are typical problems faced in this area. More recent papers attempt to solve these issues, e.g., [209].

In the *models* category, we found primary studies targeting *formal model analysis*, model-driven engineering, model reuse, and practices. Five papers focus on formal model analysis with different goals: analysis of safety and scalability in models for autonomous vehicles [185], quantitative

analysis for systems based on recurrent neural networks [50], improvement and certification of robustness for ML models [224], and approaches for formally checking safety [209] or security properties of neural networks [210]. Furthermore, four papers study the use of models as the initial step for derivation of other artefacts (*model-driven engineering*). Examples are the development of big data ML software [109] and the incorporation of safe and robust control policies in ML models [70]. The other two from the same authors [198, 199] target the derivation of testing frameworks for evaluating properties of an autonomous driving system with ML components. Lastly, we identified three primary studies addressing *model reuse* using different approaches, such as an analysis of current model reuse practices and challenges for building systems based on artificial neural networks [69], and tools to retain and reuse implementations of deep neural networks [175].

Key takeaways for SE models and methods:

- 18 papers cover the verification and validation of cyber-physical systems with AI components.
- Safety as a non-functional aspect and application domains like autonomous vehicles are very prevalent in this area.

6.8 Software Quality (59 Studies)

Quality management is a very broad area of SE, including a number of topics such as specifying quality requirements, measuring / assessing quality, and assuring quality (SWEBOk). So far, quality management during engineering of AI-based systems has been dominated by testing and formal verification methods (see sections on *testing* and *models and methods* above). Only 17 publications address the topic of defining and assessing quality of AI-based systems. Among these, two papers [60, 179] discuss software technical debt, a derivative of software quality which refers most commonly to increased costs for maintenance and evolution due to earlier quality deficits. In particular, the authors warn software engineers tempted by quick wins of data-driven software systems of forgetting that these wins are not coming for free. To avoid incurring significant technical debt in terms of ongoing maintenance costs with AI systems, the authors explore technical debt-related risks, e.g., related to a software system itself as well as to associated data and data management systems.

Furthermore, five articles propose *ML quality certification and assessment* to mitigate quality risks of deployed AI systems [12, 94, 145, 163, 224]. Several of these investigate challenges of certifying AI systems and look for potential solutions in traditional safety-critical domains such as automotive, avionics, or railway, in particular how certification approaches in these domains evolved to adjust to technological advances. Proposed certification approaches include the assessment of development processes (including workflows and engineering choices) and their impact on the quality of delivered outcomes [94]. The quality of AI systems is viewed from various perspectives, e.g., prediction performance quality, training mechanism quality, and lifecycle support quality including continuous operations [145]. Inspired by declarations of conformity – multi-dimensional fact sheets that capture and quantify various aspects of the product and its development to make it worthy of consumers' trust – authors propose that AI service providers publish similar documents containing purpose, performance, safety, security, and provenance information for their customers [12].

The most commonly discussed quality characteristics include safety and related aspects such as robustness or explainability. In addition to specific quality characteristics, meta-characteristics of AI systems, such as provability (extent to which mathematical guarantees can be provided that

some functional or non-functional properties are satisfied) or monitorability (extent to which a system provides information that allow to discriminate “correct” from “incorrect” behavior) [94], are discussed in this context as prerequisites for quality assessment and certifications. Several articles investigate quality aspects specifically relevant for AI-based systems, mostly based on important new challenges that software and requirements engineers must address when developing AI systems. Major trends in our sample are ML-specific quality aspects, such as *ML safety* [7, 32, 74, 176, 184], *ML ethics* [31, 39, 78, 114], and *ML explainability* [95, 150, 165]. Additionally, three articles from the same team of authors [116–118] discuss how individual AI quality aspects relate to each other in the context of ISO 25000 [87] as an established SE quality model. They also propose adaptations to the standard and how to quantitatively measure some AI quality aspects.

Due to the differences between AI-based systems and “traditional” software systems, six studies covered the *update of the ISO 26262 standard* to address this. Contributions range from analyzing the deficiencies of the current version of ISO 26262 [68, 106, 170, 172], to concrete adaptation proposals [79], or a methodology framework for identifying functional deficiencies during system development [37].

With 11 primary studies, *ML quality assurance frameworks* constitute another important topic. These frameworks normally focus on specific quality aspects of ML products, such as allowability, achievability, robustness, avoidability and improbability [149], safety [47, 136], specific safety issues like forward collision mitigation based on the ISO 22839 standard [57], security [51], algorithmic auditing [163], robustness diversity [189], data validation [29], or the reconciliation of product and service aspects [143]. Other approaches focus on continuous quality assurance with simulations [12] and on run-time monitoring to manage identified risks [107]. Furthermore, four primary studies explore assurance cases. Ishikawa et al. discuss the use of arguments or assurance cases for ML-based systems [89], including a framework for assessing the quality of ML components and systems [88]. Assurance cases have been also used in arguing the safety of highly automated driving functions, e.g., to solve underspecification with graphical structuring notation [66] or to address functional insufficiencies in CNN-based perception functions [32].

Four studies focus on *ML defects*, i.e., several researchers have studied the specific types of bugs in AI-based systems [37, 92, 122, 195]. Similarly, five articles discuss *ML attacks*, mostly with a focus on the use of adversarial examples [49, 55, 96, 189], for instance by applying adversarial perturbations under different physical conditions in cyber-physical systems. Tramer et al. also discuss attacks to steal the complete models of AI-based systems [197].

As with other software systems, ML-based systems require *monitoring*. We can find monitoring approaches combining ML with runtime monitoring to detect violations of system invariants in the actions’ execution policies [132], managing identified risks, catching assumption violations, and unknown unknowns as they arise in deployed systems [107], and as a runtime safety [57] or ethical [12] supervisors.

The remaining primary studies focus on either *challenges* or establishing a research *roadmap*, which is detailed in Section 7. We can highlight that in the 11 primary studies discussing roadmaps, they are often related to safety and the standard ISO 26262 [7, 68, 74, 106, 203]. This seems to be a major challenge for which people not only work on detailed research contributions but see the need for a larger research roadmap. These roadmaps usually include suggestions for extensions of the standards and V&V methods. There are two primary studies [74, 96] that also address security and attacks. One roadmap combines it with the safety roadmap and calls for better integration of ML development into SE methods. The other roadmap concentrates on different types of attacks and countermeasures. Further explicitly mentioned quality attributes for which there is a roadmap are user experience [222] and fairness [81]. One roadmap also discusses quality assurance

certification [12] and proposes to add FactSheets to ML services to increase trust. Finally, three primary studies [117, 174, 203] propose roadmaps for general quality with ML-specific extensions to the ISO 25000 standard series. They include diverse aspects such as processes, V&V methods, and formal analysis.

Key takeaways for software quality:

- The specific quality aspects of AI-based systems have triggered the need to update standards such as ISO 25000 and ISO 26262.
- Most of the studies focus on ML quality attributes, frameworks, assurance and certification.

6.9 Remaining SWEBOK Areas (4 Studies)

Other SWEBOK areas are present to a lesser extent in our sample. In the *SE management* area, Wolf et al. showed that AI software projects lead to dynamic and complex settings which necessitates active and engaged sensemaking [214]: software teams must strive to create coherence between AI environments, AI model ecosystems, and the business contexts that emerge while building AI systems. In the second study in this area, Raji et al. introduced a framework for algorithmic auditing for end-to-end support during the internal AI system development life cycle [163]. For *software configuration management*, we identified one paper: Schelter et al. presented experiences with ML model management at Amazon and outlined challenges [177]. Furthermore, one paper was categorized as *SE professional practice*. With the FactSheets approach from Arnold et al., AI service providers can publish documentation about their AI system including information on safety or data provenance to create an environment of transparency and trust [12]. Lastly, we did not identify any primary study in the area of *SE economics*, nor for *computing, mathematical, and engineering foundations*.

6.10 Discussion

The aggregated results for RQ3 imply several notable findings, which we briefly discuss in this section.

Observation 3.1: Many studies in our sample were related to software testing (115/248) and software quality (59/248). These two SWEBOK areas received particular attention, implying that their state of research is much more advanced compared to the other areas. While testing- and quality-related challenges of AI-based systems are by no means completely solved, researchers active in these areas should be especially careful when selecting the scope of their contributions and positioning them regarding existing work. More focused literature studies can support such a fine-grained overview. While several such studies exist for the testing of AI-based systems (see Section 2.2), the diverse field of software quality in this area would still benefit from a detailed review, especially since most related studies in our sample are focused on approaches related to safety, robustness, or reliability in the context of cyber-physical systems like autonomous vehicles.

Observation 3.2: The area of software maintenance is one of the smallest categories in our sample (6/248). While a few studies from software quality may also touch maintainability, evolvability, or the concept of technical debt, software maintenance received very little attention overall. Considering the peculiarities of AI-based systems and the importance of an effective and efficient maintenance and evolution for any long-living production system, this may constitute an important research gap. A reason for this could be that most researchers and practitioners still focus on the effective initial creation of these systems and may not yet have considered optimizing

their maintenance. Additionally, many AI-based systems of the new wave are not that old and may therefore not yet require sophisticated approaches necessary for, e.g., decades-old code bases.

Observation 3.3: In the SE process area, we can find recent important contributions, but rather context-specific than widely adopted ones. We believe that multidisciplinary research in this area is needed to integrate data collection and AI modeling into the SE lifecycle and vice versa. We did not find a standard process nor manifesto in the primary studies in this direction. Also, some processes mainly evolved from the data mining area (e.g., CRISP-DM), lacking an SE perspective.

Observation 3.4: Most discussed SWEBOK areas include several recent state-of-practice studies to identify concrete peculiarities for the development of AI-based systems (e.g., via surveys, interviews, StackOverflow mining, etc.). Researchers are still in the process of discovering and analyzing challenges and practices in this field, indicating that SE4AI research continues to be in a formative stage. Since finding out what techniques practitioners in this area actually use and what concrete problems they face is essential, there is still the need for additional studies like this, especially in less prevalent SWEBOK areas identified by us.

Observation 3.5: The majority of identified studies are only concerned with a single SWEBOK area (189/248). While there is much value to be gained from studies with such a detailed focus, the creation of a successful AI-based system requires an effective interplay and connection between the majority of SWEBOK areas. While some studies from the SE process area also took such a perspective, we identified only very few holistic studies in total. As the SE4AI field matures, there may be much potential for approaches that incorporate the different SE facets for AI-based systems in their entirety.

7 RQ4: WHAT ARE THE EXISTING CHALLENGES ASSOCIATED WITH SE FOR AI-BASED SYSTEMS?

As detailed in Section 3, the 39 papers that we analyzed in RQ4 included 94 challenges. A challenge may be classified into more than one SWEBOK Topic, although most of the challenges (70%) were classified under one Topic only.

Table 5 and Figure 20 summarize the number of papers, challenges and assignments to every SWEBOK Knowledge Area. Some Knowledge Areas prevale, although it cannot be said that a Knowledge Area excels significantly from the rest; furthermore, the order is different if we focus on the papers or on the challenges, because as said above several papers identify a number of challenges related to one particular Knowledge Area.

If we look at the SWEBOK Topics, we see that some of them are quite popular. Table 6 shows those referenced by four challenges or more. Remarkably, three of the topics that we proposed as an extension of SWEBOK appear in the top five positions, which is somehow natural (they naturally emerged because they were popular).

We describe below the challenges of each Knowledge Area.⁶ As for reporting style, we have opted to include the challenges respecting the words of the primary studies' authors; therefore, our work has consisted mostly in grouping and articulating these challenges into a unifying narrative. This also implies that we are intentionally refraining from adding our own interpretations or enriching in any way the challenges identified in the primary studies. Last, for readability, we neither quote the challenges nor include all citations in the text; Table 7 includes the references for every SWEBOK Knowledge Area.

⁶Names of SWEBOK Knowledge Areas and Topics appear in italics.

Table 5. Distribution of Challenges into SWEBOK Knowledge Areas

Category	#papers	#challenges	#assignments
Software Engineering Models and Methods	11	17	17
Software Quality	9	14	15
Software Construction	9	9	10
Software Testing	8	14	19
Software Design	7	10	10
Software Runtime Behaviour	6	10	10
Software Requirements	5	16	18
Software Engineering Process	5	8	12
Software Engineering Professional Practice	4	11	12
Software Maintenance	3	3	3
Software Engineering Economics	2	3	3
Software Configuration Management	1	1	1

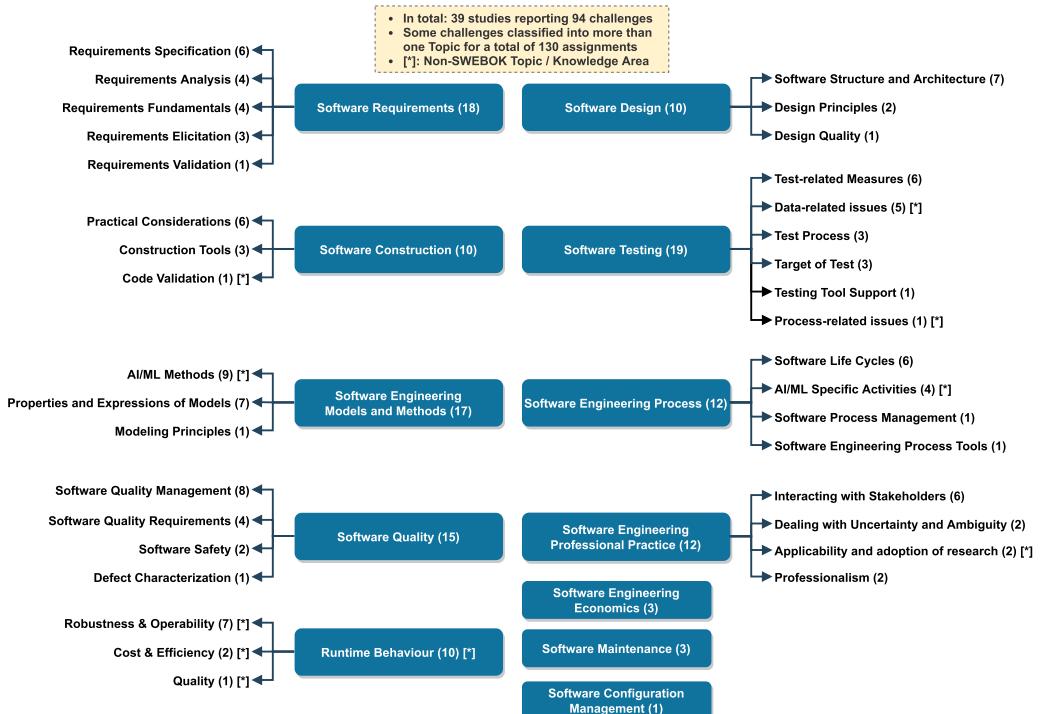


Fig. 20. Distribution of challenges into SWEBOK knowledge areas.

7.1 Software Requirements

Most of the challenges relate to one activity in the requirements engineering cycle, except a first subset related to fundamentals:

- **Fundamentals**. A number of challenges relate to functional and non-functional requirements fundamentals, arguing that: (i) our understanding of NFRs for ML is fragmented and

Table 6. Most Popular SWEBOK Topics as Referenced in the Challenges

SWEBOK Topic	SWEBOK Knowledge Areas	#challenges
AI/ML methods*	Software Engineering Models and Methods	9
Properties and expressions of models	Software Engineering Models and Methods	7
Robustness & operability*	Runtime Behaviour	7
Interacting with stakeholders	Software Engineering Professional Practice	6
Data related issues*	Software Testing	5
Software quality assessment	Software Quality	5
Software structure and architecture	Software Design	5
AI/ML specific activities*	Software Engineering Process	4
Quality requirements	Software Quality	4
Requirements specification	Software Requirements	4

* Proposed extension of SWEBOK.

Table 7. Primary Studies Containing Challenges Per SWEBOK Area

SWEBOK Knowledge Area	List of references
Software Engineering Models and Methods	[20, 64, 74, 81, 90, 94, 106, 120, 142, 177, 184]
Software Requirements	[82, 90, 94, 205, 228]
Software Testing	[28, 64, 83, 152, 174, 177, 184, 228]
Software Quality	[64, 81, 82, 90, 94, 114, 136, 174, 184]
Software Engineering Professional Practice	[86, 90, 174, 205]
Software Construction	[82, 90, 94, 117, 125, 135, 138, 174, 177, 228]
Software Runtime Behaviour	[7, 82, 120, 136, 177, 184]
Software Engineering Process	[90, 117, 123, 125, 138]
Software Design	[20, 114, 123, 125, 177, 228]
Software Maintenance	[82, 106, 177]
Software Engineering Economics	[90, 135]
Software Configuration Management	[177]

incomplete, (ii) some new NFR types need to be considered, e.g., related to explainability and freedom for discrimination, and (iii) measurement of functional requirements in practice for both functional and non-functional requirements is needed.

- *Elicitation.* The use of data as a source of requirements is attractive, but due to the high volume of such data, it requires tool support to detect features from massive data. Also, adopting the adequate stakeholder perspective is necessary, for instance to elicit explainability requirements, the user's point of view needs to be adopted. Finally, it is crucial to elicit the characteristics (e.g., related to ethnicity or gender) that must not be used in AI-based systems to avoid discriminating samples.
- *Analysis.* The most important challenge is the need to negotiate upon unfeasible 100% accuracy demands issued by customers. Moreover, it is mentioned that verifying the model features extracted from data is challenging.
- *Specification.* It is difficult to specify requirements for AI-based systems due to four main challenges: (i) transferring the problem definition into specifications, (ii) specifying clearly the concepts used by ML techniques (i.e., domain knowledge), (iii) understanding how requirements can be measured, and (iv) making clear specifications from which testing and quality assurance activities can be derived. In the case of non-functional requirements, also the challenge of defining ML-specific trade-offs is mentioned.

- *Validation.* This activity is endangered by the inherent uncertainty of the results produced by AI-based systems (e.g., relating to accuracy, cost, etc.), as well as by the difficulty to understand and use the notion of requirements coverage. Uncertainty of results has been said to be a source of anxiety for customers.

7.2 Software Design

Software design challenges occur at different levels:

- *Design principles.* It is a challenge to overcome the **CACE principle (Changing Anything Changes Everything)**, which is mainly due to the entanglement created by ML models.
- *Software structure and architecture.* Several situations particular to AI-based systems result in challenges to their structure: (i) ethics is a main issue of these applications, and a challenge is where to place the logic that governs ethical behaviour responding to criteria like scalability (needed with the advent of technologies such as 5G that demand huge amounts of sensors); (ii) AI-based systems need to deal with undeclared customers, who need to consume predictions of models; (iii) it becomes necessary to orchestrate different systems that are glued together in real-world deployments; (iv) it is requested to provide automatic exposure of ML metadata to automate and accelerate model lifecycle management; and (v) it is needed to manage the consequences of the concurrent processing required by most AI-based systems, especially those implementing DL approaches. As a particular case in this Topic, the SWEBOK sub-Topic *Design patterns* is also mentioned, given that the complexity of deploying ML techniques into production results in the emergence of several anti-patterns (glue code, pipeline jungles, etc.), making architecting a kind of plumbing rather than engineering activity.
- *Design Quality.* It is needed to reconcile conflicting forces, namely reproducibility, collaboration, ease of use and customization, in a single AI-based system.

7.3 Software Construction

Contrary to the two former Knowledge Areas, challenges reported for software construction are of a very diverse nature, most of them belonging to the *Practical Considerations* Topic:

- Implementation of ML algorithms involves several issues, among them strong dependency on data, high level of parallelism or use of complex tools like TensorFlow.
- End-to-end AI-based systems often comprise components written in different programming languages, making the management of applications challenging (e.g., ensuring consistency with error-checking tools across different languages).
- Implementation of AI-based systems with third-party components of any kind. This poses significant challenges in safety given that these components may not be assured with traditional methods, compromising their adoption in industries like avionics.
- It is a challenge to control quality during the development of DL applications.
- The integration of AI/ML components with traditional software is also mentioned as a challenge from a quality perspective.

A number of challenges are related to *Software Construction Tools*:

- Companies need to struggle with non-flexible and functionally limited AI/ML development tools that are difficult to be incorporated within the company process, resulting in fragmented toolchains that hamper the work of business analysts, who can not generate ML prototypes quickly to experiment with.

- The lack of tool infrastructure to support the development and deployment of DL solutions. This challenge is aggravated by the lack of expertise in usual IT teams to build this infrastructure.

Last, *Code Validation* was mentioned as a challenge due to the difficulty in understanding and using the notion of requirements coverage when validating the code.

7.4 Software Testing

We found some challenges directly related to *Key Issues* of software testing, majorly related to data and one also to process:

- Data and models cannot be strongly specified *a priori*, which means that testing of AI-based systems is dependent upon uncertain data and models.
- Achieving scalability of testing with millions of parameters in AI/ML applications.
- The way AI-based systems are developed, i.e., through a training phase, introduces the risk of overfitting training data.
- Dealing with the inherent incompleteness of training and testing data is a major challenge that yields to insufficiencies when the context of execution is not fully represented in the training set. This is crucial in life-critical systems as autonomous vehicles.
- Difficulty to collect enough data to test AI-based systems; for this reason, it is suggested to develop tools that augment real data while maintaining the same semantics.
- Long-running experiments and complex interactions between pipelines of models makes traceability of results' changes difficult to keep.

A number of challenges belong to the *Test-Related Measures* Topic, with special emphasis on coverage:

- The need of systematic methods to prepare quality training and coverage-oriented datasets.
- Understanding what coverage means and how to improve it are more focused challenges also reported in a couple of papers.

The rest of the challenges applies to the following topics:

- *Test Process*. The generation of reliable test oracles and effective corner cases are process activities identified as challenging. As a practical consideration, it is reported that repeatability of test results is difficult to achieve due to the ability of AI-based systems to learn over time.
- *Target of Test*. It is reported that AI-based systems suffer from what is called the oracle problem: that ground truth data is difficult or sometimes impossible to get. Furthermore, the problem of having millions of parameters mentioned above also impacts on this subcategory due to the inherent variability behind these parameters.
- *Testing Tool Support*. How to develop automatic solutions to support testing of AI-based systems.

7.5 Software Engineering Process

Most of the challenges are related to the *Software Life Cycle* Topic, as follows:

- *Life cycle models*. It is necessary to have highly iterative models that allow to evolve solutions quickly. Going further, there is a need for continuous engineering because AI-based systems can be easily invalidated by trend changes. Development iterations are slowed down when re-assessing the models after introducing changes.
- *AI/ML specific activities*. This newly proposed Topic groups all challenges around activities that are specific to AI-based systems. In particular: (i) accurate and consistent annotation

processes; (ii) ability to reproduce model selection experiments quickly; (iii) interleaving execution of experiments with interpretation of their results; and (iv) exploration of different options of ML solutions.

- *Practical considerations.* First and foremost, AI-based software development requires software engineers. More focused challenges are: (i) the need of scaling models at the end of the life cycle (production); and (ii) the difficulty of managing complex and poor logging mechanisms provided as part of the system infrastructure that makes analysis hard.

In addition, we find challenges related to:

- *Software Process Management.* It is reported the challenge of managing complex workflows in charge of learning the ML models and bringing them into production.
- *SE Process Tools.* Linked to the challenge of annotation processes, it is a need to produce annotation tools for forming accurate and consistent annotations in large datasets.

7.6 Software Engineering Models and Methods

The *Modeling* Topic has several challenges associated to its sub-topics:

- *Modeling Principles.* It is necessary to avoid model overfit, which may be produced because of either: (i) training data having accidental correlations unrelated to the desired behaviour, or (ii) validation data not independent or diverse from the training data in every way except the desired features.
- *Properties and Expressions of Models.* First, data itself poses several challenges, such as: (i) models need to rely upon high quality, properly curated datasets, as an indispensable requirement towards fairness of ML models; (ii) data should include as much as possible rare cases, which could entail learning problems, even considering that this rarity can make their collection expensive; and (iii) data volumes and variety should be enough regarding the intended provided function. Second, the model built upon this data poses several difficulties: (i) it should capture behaviour that escapes the human eye (even at the cost of making validation more complex); (ii) related to this, the model provides results that are hard (if not impossible) to understand intuitively by humans; and (iii) going further, the models are not just counter-intuitive but non-deterministic and providing uncertain predictions, and thus difficult to test. Last, the expression of the ML models is challenging due to the need of combining different models embedded therein) and also the difficulty of handling data dependencies when defining the models, which may convey different problems as instability and correction cascades, among others.
- *AI/ML methods.* We have identified this new topic that fits perfectly with the existing SE Methods' subtopics in SWEBOk (namely, heuristic methods, formal methods, prototyping methods, and agile methods). Challenges are: (i) expensiveness of data labeling; (ii) hard reasoning on robustness of ML techniques (especially worst case behaviour); (iii) development of systematic methods for preparing training and validation datasets; (iv) even in presence of these methods, ability to deal with incomplete training data; (v) model management in real-world ML deployments that involve complex orchestration frameworks or heterogeneous code bases written in different programming languages; (vi) automatic or semi-automatic formal verification of models; (vii) dealing with all aspects of data management, embracing data collection (e.g., lack of metadata), data exploration (e.g., heterogeneity of data sources), data preprocessing (e.g., cleaning of dirty data), dataset preparation (e.g., data dependencies), deployment (e.g., overfitting), and post deployment (e.g., feedback loops).

7.7 Software Quality

A first set of challenges are related to the *Software Quality Management Process* in its three subtopics:

- *Software Quality Assurance.* Challenges are manifold: (i) define quality assurance standards for AI-based systems and ensure that they scale well (e.g., they should adapt well to the continuous evolution of ML models and thus system behaviour); (ii) establish quality assurance criteria in the presence of big data (as required by AI-based systems); (iii) dealing with not assured components (e.g., third-party components or legacy software); and (iv) assurance of safety and stability is particularly challenging because there are no clear principles established.
- *Verification & Validation.* In general, verification and validation of the model produced by the training process regarding the intended function is challenging, principally because it needs to recognize the fact that the output of an ML model responding to a given input cannot be completely predicted.
- *Reviews and Audits.* Fairness auditing is threatened by the fact that it requires collecting information at individual-level demographics, which is rarely possible; new methods need to adapt to this reality and allow demographics at coarser levels.

A similar number of challenges appears concerning *Practical Considerations of Software Quality Requirements*:

- Ensuring that the AI-based systems will not reinforce existing discriminations (on gender, race, or religion).
- Dealing with the limited knowledge of the effects that AI-based systems have on quality requirements (including their trade-offs).
- Considering runtime quality when analysing the effects of AI-based systems on quality requirements.

An additional practical consideration emerges in the *Defect Characterization* subtopic: the difficulty to explain to customers the failure in making a certain output due to the black-box nature of AI-based systems, especially when the real system output is counter-intuitive.

The last two challenges were specifically related to *Software Safety*: (i) as stated above, absence of principles for quality of safety; (ii) oversimplification of safety assessment, not considering that it needs to remain valid through the application lifetime.

7.8 Software Engineering Professional Practice

We grouped the challenges related to professional practice into the following topics:

- *Interacting with Stakeholders.* (i) We find challenges especially related with understanding of, and interaction with the customer because customers may have unrealistic expectations regarding the functionality, accuracy (requiring even 100% accuracy), or adoption process of AI-based systems (expecting solutions to start to work with too little data available). At the end, it is necessary for the data scientists' team to have the skills to interact with the customers and help them to set reasonable targets. (ii) From a more practical standpoint, due to the inherent iterative nature of AI-based systems that require continuous improvement of solutions, it becomes necessary to convince the customer about the need to keep paying continuously. (iii) Due to the blackbox nature of AI-based systems, it is a challenge to explain the customer failure to produce expected results.

- *Applicability of Research in Practice.* We propose this new subcategory to group challenges related to transfer of research. Probably the most typical one (not only in the AI domain) is the oversimplification of reality when it comes to developing ML models. Nowadays, complex systems as autonomous systems have thousands of sensors and run several programs together, therefore toy academic examples (e.g., “Lego Mindstorms”) are not acceptable in real settings. Also, it is mentioned the impediment of the risk perception of AI/ML results from the general public, which also needs to be considered when developing realistic AI-based systems.
- *Dealing with Uncertainty and Ambiguity.* Running ML/AI projects in real environments requires the ability to deal with the uncertainty of both estimating development time and cost, and validating the application considering that there is not a well-defined factor from any possible input to a given output.
- *Code of Ethics and Legal Issues.* Professional practice requires to consider these two aspects, which are challenging considering the intensive use of data by ML models (e.g., compliance to GDPR when dealing with personal data).

7.9 Software Runtime Behaviour

As explained in Section 3 and 7, we added this Knowledge Area that is not proposed in SWEBOK due to the importance of this aspect in AI-based systems. We further distinguished the following Topics:

- *Robustness and Operability.* It refers to the fact that an AI-based system must be robust and easy to operate while in use. In this respect, mentioned challenges are: (i) the need to avoid negative side effects, so that the behaviour of the application does not damage its environment; (ii) in a similar vein, ensuring safe exploration during the learning process; (iii) preventing the hacking or gaming of reward functions, which could lead to artificially considering that the application reached its objectives when it is not true; (iv) adaptation of the application responding to changes in the operational environment; (v) dealing with unpredictable behaviour when the input of the application does not completely align with the training set (“distributional shift”); and (vi) coordinating the workloads of the different systems that compose an ML pipeline at runtime.
- *Cost and Efficiency.* It tackles the problem of achieving efficient and cost-effective behaviour at runtime. We found two challenges: (i) provide scalable oversight for tasks for which there is insufficient information (by involving the human in the loop); and (ii) overcome stringent timing and energy constraints which conflict with the resource-intensive nature of ML/AI applications.
- *Quality.* A challenge is to understand the effects of ML algorithms on desired qualities not only during ML solution design, but at runtime – during the lifetime of the ML solution.

7.10 Remaining SWEBOK Areas

A handful of challenges related to some remaining SWEBOK Knowledge Areas:

- *Software Maintenance.* All the challenges were related to training and validation data: (i) it is difficult to determine the frequency of retraining the models because training is usually conducted offline, therefore if there are changes in the context not captured by the training data, the model may become outdated and retraining is needed; and (ii) as a follow-up of the preceding challenge, even minor changes on the training data may provoke a radical change in the learned rules, requiring thus complete revalidation of the model.

- *Software Configuration Management*. A main impediment to Software Building is the need of orchestrating different systems to deploy AI-based systems in a real context.
- *SE Economics*. Three challenges were reported relating to *Risk and Uncertainty* and *Value Management*: (i) how to quantify and assess the value of the knowledge generated in the company in the process of developing an AI-based system (even in the case that the application does not reach the initial goals); (ii) how to assess the long-term potential value of an AI/ML prototype, with the short-term metrics that can be usually gathered; and (iii) how to manage the impossibility to make any prior guarantee on cost-effectiveness of AI-based systems due to uncertainty of their behaviour.

7.11 Discussion

Observation 4.1: Challenges are highly specific to the AI/ML domain. Not only have we defined a non-SWEBOK Knowledge Area for covering runtime-related aspects, but also, we have classified in total 32 out of the 130 instances of challenges (i.e., 24.6%) into non-SWEBOK Topics. This result provides evidence about the specificity of challenges reported by researchers and practitioners when it comes to AI-based systems and AI/ML model development. Another related observation is that artefacts that are widespread in the SE community as the SWEBOK body of knowledge are not totally fit to the AI-based systems engineering. Other artefacts which may suffer from similar drawbacks are quality standards such as ISO 25010 and best practices such as architectural or design patterns.

Observation 4.2: Challenges are mainly of a technical nature. In general, more technical Knowledge Areas are those with more challenges identified, with the only exception of *SE Professional Practice*, which in fact appeared as an accompanying Knowledge Area in five cases (in the same paper). Even in this Knowledge Area, the Topics identified were mainly two, namely *Interacting with Stakeholders* and *Dealing with Uncertainty and Ambiguity*, while other fundamental topics as *Legal Issues*, *Codes of Ethics*, or *Standards* were only accidentally mentioned or not mentioned at all. We may add the little importance given to *SE Economics challenges*, with Topics such as *Risk*, *Return on Investment*, or *Replacement and Retirement Decisions* not mentioned at all. Moreover, we did not identify challenges for the three foundations Knowledge Areas (computing, mathematical, and engineering), although in this case, they may be hidden behind the technical challenges reported.

Observation 4.3: Data-related issues are the most recurrent type of challenge. Digging further into the technical nature of the challenges, we see a dominance of issues related to data, from the different perspectives provided by SWEBOK's Knowledge Areas. We find challenges related to different stages of the software process (e.g., identifying features over a large amount of data during requirements elicitation, preparing high-quality training datasets during testing), and also to transversal activities such as quality management (e.g., effects of data incompleteness on the overall system quality). In contrast, the surveyed papers have identified very few mitigation actions to cope with these challenges, e.g., generation and simulation of rare cases data to manage edge cases during learning. Therefore, the lack of such mitigation actions constitute research gaps to be addressed. This last observation aligns with other studies, e.g., Lwakatare et al. [2020] only presents eight solutions for the 23 challenges that they identify in their paper.

8 THREATS TO VALIDITY

As any other empirical study, ours faces a series of threats to validity. We report them below according to frequent threats specific to secondary studies [8] including mitigation actions. These specific threats are divided into three categories: study selection validity, data validity, and research validity. As a global mitigation action, we have considered in our study the ACM SIGSOFT

Empirical Standards [164], and in particular we have ensured: (i) to comply with all the eight essential specific attributes for systematic reviews; and (ii) to avoid the three anti-patterns that apply to systematic reviews (i.e., not synthesising findings, not including quality assessment of primary studies, and shortage of high-quality primary studies).

8.1 Study Selection Validity

Study selection threats can be identified in the steps 1 to 3 of our SMS, mainly conducting the search and screening of papers.

One of the inherent threats to any SMS is that it does not guarantee the inclusion of all the relevant works in the field. To mitigate this threat, we first conducted multiple pilot tests to build the search string to obtain the seed papers. Details of the different search strings options are available in our replication package. Four researchers independently undertook the preliminary search process before finalizing the search scope and search keywords. In a meeting with all researchers, initial search strings were discussed, and the reported one in Section 3 selected.

To mitigate sampling bias and publication bias, we used a combination of: (i) keyword automated searchers for seed papers and manual snowballing; (ii) checking the annotated bibliography of an external prolific researcher; and (iii) searching in multiple indexes: Scopus for seed papers, and Google Scholar for snowballing considering pre-print servers (i.e., arXiv). We applied iteratively backward and forward snowballing until we reached 248 primary studies, which ensured that we covered a significant list of papers in a wide range of SE topics. As the field of SE4AI is further emerging and our resources for this study are limited, we decided to stop at this point.

By following the snowballing strategy, we also mitigated any possible threat related to the lack of a standard use of terminology, or lack of any relevant term in the search string, which is a common threat to validity in a pure string-search based methodology.

During the primary studies selection process, to mitigate any possible bias when applying the inclusion/exclusion criteria, these criteria were defined and updated in our protocol. Furthermore, each paper was assigned to two researchers from different institutions to decide about its inclusion or exclusion. Any disagreement was discussed between the two researchers, and if an agreement between the two was not reached, a third researcher was involved to make the final decision.

Regarding quality assessment, we used a classification of research types, including those without empirical evidence. Articles in this category would generally not be included in a systematic literature review, though in SMS they are important to spot trends of topics being worked on. We reported this in Section 4, and considered all papers as the outcome of an SMS is an inventory of papers on the topic area, mapped to a classification [159]. Furthermore, we have also extracted information about the research rigor and industrial relevance of the primary studies.

8.2 Data Validity

Data validity threats can be identified in the steps 4 and 5 of our SMS: key wording, and data extraction and mapping process.

During the process of data extraction, subjective bias may lead to the misclassification of data or an inconsistent interpretation of the extracted data by the researchers. To mitigate these risks, we piloted the data extraction form, conducted weekly meetings with all the researchers, and discussed potential issues related to data extraction. All found issues were discussed among all researchers and decisions were documented to ensure that all researchers followed consistent data extraction and synthesis criteria. Nonetheless, apart from the three studies used for piloting, each paper was extracted by a single researcher. While many extractions were fairly objective (e.g., a paper either explicitly described threats to validity or not, a paper used a specific term for AI-based system or a quality attribute goal, etc.), others left more room for interpretation. However, we

argue that complete agreement is neither attainable for a sufficiently complex extraction process with multiple researchers nor is it strictly necessary, since we are very confident in the general tendencies and take-aways based on the extracted and synthesized data.

Furthermore, as explained in Section 3, we performed qualitative analysis through an existing conceptual framework (SWEBOK). We iterated on initial classifications among all researchers in our weekly meetings, leading to some proposals to update this framework.

Lastly, we need to mention that we adopted an inductive approach to the coding of properties. During the data extraction and mapping process, we e.g., extracted quality attribute goals and then grouped similar terms into unique codes. Including such terms explicitly in the search string may have produced slightly different results. Overall, we are confident that snowballing led to valid general tendencies in our sample, even though we do not claim completeness.

8.3 Research Validity

Threats that can be identified in all steps of our SMS are classified as research validity.

We have performed a broad search of secondary studies (see Section 2.2). This allowed us to understand research gaps and brainstorm about the coverage and definition of our RQs. We have compared the results of our SMS to the current state-of-the-art within the scope of SE4AI, to generalize our findings with this scope.

To enable the reproducibility by other researchers of this SMS, we have documented all the steps performed, along with all the intermediate results. We have described the procedure in detail in Section 3. Furthermore, all the raw materials and documented process are available in our replication package.

9 CONCLUSIONS

In this paper, we surveyed the literature for software engineering for artificial intelligence (SE4AI) in the context of the new wave of AI. In the last ten years, the number of papers published in the area of SE4AI has strongly increased. There were almost no papers up to 2015 while afterwards, we saw a strong increase to 102 in 2019. The share of more than 18% on arXiv shows the “hotness” of the topic, but also emphasizes that literature reviews need to take arXiv into account. Furthermore, most articles are from a purely academic context, but 20% of publications with only industry authors show the importance for practice. When we look at the countries of the authors, the United States plays in a separate league altogether, while China, Germany, and Japan are the strongest of the remaining countries.

The empirical studies in our sample seem to form a healthy mix of case studies, experiments, and benchmarks. The latter play a larger role than in other fields of SE, which can be explained by the data-driven nature of the methods that often lend themselves to being benchmarked. In these studies, we see overall many realistic problems, data sets, and applications in practice. The involvement of practitioners improves some quality characteristics of the studies, like the realism of the case studies, and more significantly, their scale. We have also found, however, that authors often ignore discussing threats to validity.

The terminology in the primary studies is all but homogeneous. This makes it often difficult to judge the scope of the contributions. We therefore propose to include a taxonomy in each SE4AI paper that clarifies the level of AI that the contribution is associated with. Furthermore, we suggest using the term *AI component* if the article is about a part of a system that uses AI. An *AI-based system* is a system consisting of various software and potentially other components with at least one AI component. Most of our primary studies are about AI-based systems or AI components. The most mentioned application domain for AI-based systems is automotive, while almost half of

the contributions are not addressing any specific application domain. In terms of methods, almost all contributions use ML techniques, with DL as the largest explicitly mentioned technique.

Regarding the SE areas the primary studies contribute to, *software testing* (115 studies) and *software quality* (59 studies) are the most prevalent in our sample. Here, the main contributions are test cases for AI-based systems testing, and the need for an update of current quality standards (e.g., ISO 26262) regarding AI-based systems. To a lesser extent, SE models and methods, SE processes, and software design have been investigated by more than 30 studies each. However, *software construction*, *software requirements*, and especially *software maintenance* are less represented and seem to offer a lot of potential for research. Examples of studies in these areas are synthesizing best practices for AI-based systems construction, as well as holistic views for requirements engineering or debugging AI-based systems. Additionally, we identified several recent state-of-practice studies in every major SWEBOK area and very few holistic approaches spanning several areas, both of which seem to indicate that the SE4AI research field is still in a formative stage.

Challenges related to AI-based systems are mainly characterized by two facts. First, a significant share of identified challenges (25%) are strongly tied to the system domain and thus difficult to classify using the SWEBOK Knowledge Areas and Topics. Second, the mentioned challenges focus mostly on technical issues, especially data-related ones, and hardly consider challenges related to other areas such as economics.

We believe this is the most comprehensive survey of the SE4AI area for the new wave of AI. Yet, as the research field is still forming, updates to this comprehensive survey as well as more focused surveys for specific domains or SWEBOK areas will be needed. Moreover, for future work, we also want to survey the SE4AI field for earlier waves of AI to compare previous approaches with current research.

APPENDICES

A DATA AVAILABILITY

The replication package contains the instruments used during the SMS: from the search string used to derive the start set of paper, to the data extraction form, and to the detailed classifications from the data analysis. It is available on <https://doi.org/10.6084/m9.figshare.14538324>.

B CREDIT AUTHOR STATEMENT

Silverio Martínez-Fernández: Conceptualization, Methodology, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization, Project administration

Justus Bogner: Conceptualization, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization

Xavier Franch: Conceptualization, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization

Marc Oriol: Conceptualization, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization

Julien Siebert: Conceptualization, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization

Adam Trendowicz: Conceptualization, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization

Anna Maria Vollmer: Conceptualization, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization

Stefan Wagner: Conceptualization, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization.

ACKNOWLEDGMENTS

We thank Andreas Jedlitschka for his feedback on earlier stages of this work and for being supportive and useful to enhance it.

REFERENCES

- [1] Raja Ben Abdessalem, Shiva Nejati, Lionel C. Briand, and Thomas Stifter. 2018. Testing vision-based control systems using learnable evolutionary algorithms. In *Proceedings of the 40th International Conference on Software Engineering*. ACM, New York, NY, USA, 1016–1026. <https://doi.org/10.1145/3180155.3180160>
- [2] Morayo Adedjouma, Gabriel Pedroza, and Boutheina Bannour. 2018. Representative safety assessment of autonomous vehicle for public transportation. In *2018 IEEE 21st International Symposium on Real-Time Distributed Computing (ISORC)*. IEEE, 124–129. <https://doi.org/10.1109/ISORC.2018.00025>
- [3] Aniya Aggarwal, Pranay Lohia, Seema Nagar, Kuntal Dey, and Diptikalyan Saha. 2019. 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*. ACM, 625–635. <https://doi.org/10.1145/3338906.3338937>
- [4] Rama Akkiraju, Vibha Sinha, Anbang Xu, Jalal Mahmud, Pritam Gundecha, Zhe Liu, Xiaotong Liu, and John Schumacher. 2018. Characterizing machine learning process: A maturity framework. *arXiv* (2018).
- [5] Mohannad Alahdab and Gülcilkh. 2019. Empirical analysis of hidden technical debt patterns in machine learning software. In *Product-Focused Software Process Improvement*. Springer International Publishing, 195–202. https://doi.org/10.1007/978-3-030-35333-9_14
- [6] 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. <https://doi.org/10.1109/ICSE-SEIP.2019.00042>
- [7] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in AI safety. *arXiv* 277, 2003 (2016), 1–29. arXiv:1606.06565 <http://arxiv.org/abs/1606.06565>
- [8] Apostolos Ampatzoglou, Stamatia Bibi, Paris Avgeriou, Marijn Verbeek, and Alexander Chatzigeorgiou. 2019. Identifying, categorizing and mitigating threats to validity in software engineering secondary studies. *Information and Software Technology* 106 (2019), 201–230.
- [9] Adina Aniculaesei, Jörg Grieser, Andreas Rausch, Karina Rehfeldt, and Tim Warnecke. 2018. Towards a holistic software systems engineering approach for dependable autonomous systems. In *Proceedings of the 1st International Workshop on Software Engineering for AI in Autonomous Systems*. ACM, 23–30. <https://doi.org/10.1145/3194085.3194091>
- [10] Adina Aniculaesei, Jörg Grieser, Andreas Rausch, Karina Rehfeldt, and Tim Warnecke. 2019. Graceful degradation of decision and control responsibility for autonomous systems based on dependability cages. *5th International Symposium on Future Active Safety Technology toward Zero Accidents (FAST-zero'19)* September (2019), 1–6.
- [11] Gary Anthes. 2017. Artificial intelligence poised to ride a new wave. *Commun. ACM* 60, 7 (2017), 19–21.
- [12] M. Arnold, R. K. E. Bellamy, M. Hind, S. Houde, S. Mehta, A. Mojsilović, R. Nair, K. Natesan Ramamurthy, D. Reimer, A. Olteanu, D. Piorkowski, J. Tsay, and K. R. Varshney. 2018. FactSheets: Increasing trust in AI services through supplier's declarations of conformity. *arXiv* (2018). arXiv:1808.07261.
- [13] Anders Arpteg, Bjorn Brinne, Luka Crnkovic-Friis, and Jan Bosch. 2018. Software engineering challenges of deep learning. In *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 50–59. <https://doi.org/10.1109/SEAA.2018.00018> arXiv:1810.12034.
- [14] Peter Bailis, Kunle Olukotun, Christopher Ré, and Matei Zaharia. 2017. Infrastructure for usable machine learning: The Stanford DAWN project. *arXiv* (2017). arXiv:1705.07538.
- [15] Alec Banks and Rob Ashmore. 2019. Requirements assurance in machine learning. *CEUR Workshop Proceedings* 2301 (2019).
- [16] Somil Bansal and Claire J. Tomlin. 2018. Control and safety of autonomous vehicles with learning-enabled components. In *Safe, Autonomous and Intelligent Vehicles*. Springer International Publishing, 57–75. https://doi.org/10.1007/978-3-319-97301-2_4
- [17] V. Basili, G. Caldiera, and H. D. Rombach. 1994. The goal question metric approach. In *Encyclopedia of Software Engineering*, Vol. 2. John Wiley & Sons, 528–532.
- [18] Denis Baylor, Eric Breck, Heng-Tze Cheng, Noah Fiedel, Chuan Yu Foo, Zakaria Haque, Salem Haykal, Mustafa Ispir, Vihan Jain, Levent Koc, Chiu Yuen Koo, Lukasz Lew, Clemens Mewald, Akshay Naresh Modi, Neoklis Polyzotis, Sukriti Ramesh, Sudip Roy, Steven Euijong Whang, Martin Wicke, Jarek Wilkiewicz, Xin Zhang, and Martin Zinkevich. 2017. TFX: A TensorFlow-based production-scale machine learning platform. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1387–1395. <https://doi.org/10.1145/3097983.3098021>

- [19] Woubshet Behutiye, Pertti Karhapää, Lidia López, Xavier Burgués, Silverio Martínez-Fernández, Anna Maria Vollmer, Pilar Rodríguez, Xavier Franch, and Markku Oivo. 2020. Management of quality requirements in agile and rapid software development: A systematic mapping study. *Information and Software Technology* 123 (2020), 106225.
- [20] Hrvoje Belani, Marin Vukovic, and Zeljka Car. 2019. Requirements engineering challenges in building AI-based complex systems. In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*. IEEE, 252–255. <https://doi.org/10.1109/REW.2019.00051>
- [21] Lucas Bernardi, Themistoklis Mavridis, and Pablo Estevez. 2019. 150 successful machine learning models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1743–1751. <https://doi.org/10.1145/3292500.3330744>
- [22] Jan Aike Bolte, Andreas Bär, Daniel Lipinski, and Tim Fingscheidt. 2019. Towards corner case detection for autonomous driving. *arXiv* Iv (2019).
- [23] 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).
- [24] Jan Bosch, Ivica Crnkovic, and Helena Holmström Olsson. 2020. Engineering AI systems: A research agenda. *arXiv* (2020). arXiv:[2001.07522](https://arxiv.org/abs/2001.07522).
- [25] Pierre Bourque and E. Richard. 2014. SWEBOK Version 3.0. *IEEE, ISBN-10: 0-7695-5166-1* (2014).
- [26] Josip Bozic and Franz Wotawa. 2018. Security testing for chatbots. In *Testing Software and Systems*. Springer International Publishing, 33–38. https://doi.org/10.1007/978-3-319-99927-2_3
- [27] Houssem Ben Braiek and Foutse Khomh. 2020. On testing machine learning programs. *Journal of Systems and Software* 164 (2020), 110542.
- [28] Eric Breck, Shangqing Cai, Eric Nielsen, Michael Salib, and D. Sculley. 2017. The ML test score: A rubric for ML production readiness and technical debt reduction. In *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 1123–1132. <https://doi.org/10.1109/BigData.2017.8258038>
- [29] Eric Breck, Neoklis Polyzotis, Sudip Roy, Steven Euijong Whang, and Martin Zinkevich. 2019. Data validation for machine learning. *SysML* (2019), 1–14.
- [30] Pearl Brereton, Barbara A. Kitchenham, David Budgen, Mark Turner, and Mohamed Khalil. 2007. Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software* 80, 4 (Apr 2007), 571–583. <https://doi.org/10.1016/j.jss.2006.07.009>
- [31] Joanna Bryson and Alan Winfield. 2017. Standardizing ethical design for artificial intelligence and autonomous systems. *Computer* 50, 5 (May 2017), 116–119. <https://doi.org/10.1109/MC.2017.154>
- [32] Simon Burton, Lydia Gauerhof, and Christian Heinemann. 2017. Making the case for safety of machine learning in highly automated driving. In *Lecture Notes in Computer Science*. Springer International Publishing, 5–16. https://doi.org/10.1007/978-3-319-66284-8_1
- [33] Taejoon Byun, Vaibhav Sharma, Abhishek Vijayakumar, Sanjai Rayadurgam, and Darren Cofer. 2019. Input prioritization for testing neural networks. In *2019 IEEE International Conference on Artificial Intelligence Testing (AITest)*. IEEE, 63–70. <https://doi.org/10.1109/AITest.2019.0000-6> arXiv:[1901.03768](https://arxiv.org/abs/1901.03768).
- [34] Shangqing Cai, Eric Breck, Eric Nielsen, Michael Salib, and D. Sculley. 2016. TensorFlow debugger: Debugging dataflow graphs for machine learning. In *Proceedings of the Reliable Machine Learning in the Wild - NIPS 2016 Workshop* (2016). <https://storage.googleapis.com/pub-tools-public-publication-data/pdf/45789.pdf>.
- [35] Aleksandar Chakarov, Aditya Nori, Sriram Rajamani, Shayak Sen, and Deepak Vijaykeerthy. 2016. Debugging machine learning tasks. *arXiv* (2016), 1–29. arXiv:[1603.07292](https://arxiv.org/abs/1603.07292) [http://arxiv.org/abs/1603.07292](https://arxiv.org/abs/1603.07292).
- [36] Anand Chakravarty. 2010. Stress testing an AI based web service: A case study. In *2010 Seventh International Conference on Information Technology: New Generations*. IEEE, 1004–1008. <https://doi.org/10.1109/ITNG.2010.149>
- [37] Meng Chen, Andreas Knapp, Martin Pohl, and Klaus Dietmayer. 2018. Taming functional deficiencies of automated driving systems: A methodology framework toward safety validation. In *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 1918–1924. <https://doi.org/10.1109/IVS.2018.8500679>
- [38] Chih-Hong Cheng, Georg Nürenberg, Chung-Hao Huang, and Harald Ruess. 2018. Verification of binarized neural networks via inter-neuron factoring. In *Lecture Notes in Computer Science*. Springer International Publishing, 279–290. https://doi.org/10.1007/978-3-030-03592-1_16 arXiv:[arXiv:1710.03107v2](https://arxiv.org/abs/1710.03107v2).
- [39] D. L. Coates and A. Martin. 2019. An instrument to evaluate the maturity of bias governance capability in artificial intelligence projects. *IBM Journal of Research and Development* 63, 4/5 (Jul 2019), 7:1–7:15. <https://doi.org/10.1147/JRD.2019.2915062>
- [40] Ricardo Colomo-Palacios. 2019. *Towards a Software Engineering Framework for the Design, Construction and Deployment of Machine Learning-Based Solutions in Digitalization Processes*. 343–349.

- [41] Dolors Costal, Carles Farré, Xavier Franch, and Carme Quer. 2021. How tertiary studies perform quality assessment of secondary studies in software engineering. In *2021 Proceedings of 24th IberoAmerican Conference on Software Engineering (CIBSE 2021), ESELAW track*.
- [42] Daniel Crankshaw, Xin Wang, Giulio Zhou, Michael J. Franklin, Joseph E. Gonzalez, and Ion Stoica. 2017. Clipper: A low-latency online prediction serving system. *Proceedings of the 14th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2017* (2017), 613–627. arXiv:1612.03079.
- [43] Daniela S. Cruzes and Tore Dyba. 2011. Recommended steps for thematic synthesis in software engineering. In *2011 International Symposium on Empirical Software Engineering and Measurement*. IEEE, 275–284.
- [44] Elizamary de Souza Nascimento, Iftekhar Ahmed, Edson Oliveira, Marcio Piedade Palheta, Igor Steinmacher, and Tayana Conte. 2019. Understanding development process of machine learning systems: Challenges and solutions. In *2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE, 1–6. <https://doi.org/10.1109/ESEM.2019.8870157>
- [45] Ryan M. Deak and Jonathan H. Morra. 2018. Aloha: A machine learning framework for engineers. *Conference on Systems and Machine Learning (MLSys)* (2018), 17–19. <https://www.sysml.cc/doc/13.pdf>.
- [46] Li Deng. 2018. Artificial intelligence in the rising wave of deep learning: The historical path and future outlook [perspectives]. *IEEE Signal Processing Magazine* 35, 1 (2018), 180–177.
- [47] Ankush Desai, Shromona Ghosh, Sanjit A. Seshia, Natarajan Shankar, and Ashish Tiwari. 2019. SOTER: A runtime assurance framework for programming safe robotics systems. In *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 138–150. <https://doi.org/10.1109/DSN.2019.00027> arXiv:1808.07921.
- [48] Tommaso Dreossi, Daniel J. Fremont, Shromona Ghosh, Edward Kim, Hadi Ravanbakhsh, Marcell Vazquez-Chanlatte, and Sanjit A. Seshia. 2019. VerifAI: A toolkit for the formal design and analysis of artificial intelligence-based systems. In *Computer Aided Verification*, Isil Dillig and Serdar Tasiran (Eds.). Springer International Publishing, Cham, 432–442.
- [49] Tommaso Dreossi, Somesh Jha, and Sanjit A. Seshia. 2018. Semantic adversarial deep learning. *arXiv* 2 (2018), 3–26.
- [50] Xiaoning Du, Xiaofei Xie, Yi Li, Lei Ma, Yang Liu, and Jianjun Zhao. 2019. 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*. ACM, 477–487. <https://doi.org/10.1145/3338906.3338954>
- [51] Xiaoning Du, Xiaofei Xie, Yi Li, Lei Ma, Yang Liu, and Jianjun Zhao. 2019. A quantitative analysis framework for recurrent neural network. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 1062–1065. <https://doi.org/10.1109/ASE.2019.00102>
- [52] Anurag Dwarakanath, Manish Ahuja, Samarth Sikand, Raghatham M. Rao, R. P. Jagadeesh Chandra Bose, Neville Dubash, and Sanjay Podder. 2018. Identifying implementation bugs in machine learning based image classifiers using metamorphic testing. In *Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ACM, 118–128. <https://doi.org/10.1145/3213846.3213858> arXiv:1808.05353.
- [53] Khaled El Emam. 1999. Benchmarking Kappa: Interrater agreement in software process assessments. *Empir. Softw. Eng.* 4, 2 (1999), 113–133.
- [54] Hasan Ferit Eniser, Simos Gerasimou, and Alper Sen. 2019. DeepFault: Fault localization for deep neural networks. In *Fundamental Approaches to Software Engineering*. Springer International Publishing, 171–191. https://doi.org/10.1007/978-3-030-16722-6_10 arXiv:1902.05974.
- [55] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. 2017. Robust physical-world attacks on deep learning models. *arXiv* (2017). arXiv:1707.08945 <http://arxiv.org/abs/1707.08945>.
- [56] Yang Feng, Qingkai Shi, Xinyu Gao, Jun Wan, Chunrong Fang, and Zhenyu Chen. 2020. DeepGini: Prioritizing massive tests to enhance the robustness of deep neural networks. In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ACM, 177–188. <https://doi.org/10.1145/3395363.3397357> arXiv:1903.00661.
- [57] Patrik Feth, Daniel Schneider, and Rasmus Adler. 2017. A conceptual safety supervisor definition and evaluation framework for autonomous systems. In *Lecture Notes in Computer Science*. Springer International Publishing, 135–148. https://doi.org/10.1007/978-3-319-66266-4_9
- [58] Rebecca Fiebrink, Perry R. Cook, and Dan Trueman. 2011. Human model evaluation in interactive supervised learning. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems - CHI'11*. ACM Press, 147–156. <https://doi.org/10.1145/1978942.1978965>
- [59] Ilias Flaounas. 2017. Beyond the technical challenges for deploying machine learning solutions in a software company. *arXiv* (2017). arXiv:1708.02363.
- [60] Harald Foidl, Michael Felderer, and Stefan Biffl. 2019. Technical debt in data-intensive software systems. In *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 338–341. <https://doi.org/10.1109/SEAA.2019.00008> arXiv:1905.13455.

- [61] Oscar Franco-Bedoya, David Ameller, Dolors Costal, and Xavier Franch. 2017. Open source software ecosystems: A systematic mapping. *Information and Software Technology* 91 (2017), 160–185.
- [62] Daniel J. Fremont, Edward Kim, Yash Vardhan Pant, Sanjit A. Seshia, Atul Acharya, Xantha Bruso, Paul Wells, Steve Lemke, Qiang Lu, and Shalin Mehta. 2020. Formal scenario-based testing of autonomous vehicles: From simulation to the real world. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. <https://doi.org/10.1109/ITSC45102.2020.9294368> arXiv:2003.07739.
- [63] Alessio Gambi, Marc Mueller, and Gordon Fraser. 2019. Automatically testing self-driving cars with search-based procedural content generation. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ACM, 273–283. <https://doi.org/10.1145/3293882.3330566>
- [64] Jerry Gao, Chuanqi Tao, Dou Jie, and Shengqiang Lu. 2019. Invited paper: What is AI software testing? and why. In *2019 IEEE International Conference on Service-Oriented System Engineering (SOSE)*. IEEE, 27–36. <https://doi.org/10.1109/SOSE.2019.00015>
- [65] Alvaro Lopez Garcia, Jesus Marco De Lucas, Marica Antonacci, Wolfgang Zu Castell, Mario David, Marcus Hardt, Lara Lloret Iglesias, Germen Molto, Marcin Plociennik, Viet Tran, Andy S. Alic, Miguel Caballer, Isabel Campos Plasencia, Alessandro Costantini, Stefan Dlugolinsky, Doina Cristina Duma, Giacinto Donvito, Jorge Gomes, Ignacio Heredia Cacha, Keiichi Ito, Valentin Y. Kozlov, Giang Nguyen, Pablo Orviz Fernandez, Zdenek Sustr, and Pawel Wolniewicz. 2020. A cloud-based framework for machine learning workloads and applications. *IEEE Access* 8 (2020), 18681–18692. <https://doi.org/10.1109/ACCESS.2020.2964386>
- [66] Lydia Gauerhof, Peter Munk, and Simon Burton. 2018. Structuring validation targets of a machine learning function applied to automated driving. In *Developments in Language Theory*. Springer International Publishing, 45–58. https://doi.org/10.1007/978-3-319-99130-6_4
- [67] Simos Gerasimou, Hasan Ferit Eniser, Alper Sen, and Alper Cakan. 2020. Importance-driven deep learning system testing. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Companion Proceedings*. ACM, 322–323. <https://doi.org/10.1145/3377812.3390793> arXiv:2002.03433.
- [68] Mohamad Gharib, Paolo Lollini, Marco Botta, Elvio Amparore, Susanna Donatelli, and Andrea Bondavalli. 2018. On the safety of automotive systems incorporating machine learning based components: A position paper. In *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, 271–274. <https://doi.org/10.1109/DSN-W.2018.00074>
- [69] Javad Ghofrani, Ehsan Kozegar, Arezoo Bozorgmehr, and Mohammad Divband Soorati. 2019. Reusability in artificial neural networks. In *Proceedings of the 23rd International Systems and Software Product Line Conference Volume B - SPLC'19*. ACM Press. <https://doi.org/10.1145/3307630.3342419>
- [70] Shromona Ghosh, Hadi Ravanbakhsh, and Sanjit A. Seshia. 2019. Counterexample-guided synthesis of perception models and control. *arXiv* (2019). arXiv:1911.01523.
- [71] 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. <https://doi.org/10.1016/j.jss.2021.111031>
- [72] Divya Gopinath, Guy Katz, Corina S. Pasareanu, and Clark Barrett. 2017. DeepSafe: A data-driven approach for checking adversarial robustness in neural networks. *arXiv* (2017). arXiv:1710.00486.
- [73] Qianyu Guo, Sen Chen, Xiaofei Xie, Lei Ma, Qiang Hu, Hongtao Liu, Yang Liu, Jianjun Zhao, and Xiaohong Li. 2019. An empirical study towards characterizing deep learning development and deployment across different frameworks and platforms. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 810–822. <https://doi.org/10.1109/ASE.2019.00080> arXiv:1909.06727.
- [74] Gaetan 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. <https://doi.org/10.1109/SYSCON.2018.8369576>
- [75] Malay Haldar, Mustafa Abdool, Prashant Ramanathan, Tao Xu, Shulin Yang, Huizhong Duan, Qing Zhang, Nick Barrow-Williams, Bradley C. Turnbull, Brendan M. Collins, and Thomas Legrand. 2018. Applying deep learning to Airbnb search. *arXiv* (2018), 1927–1935.
- [76] Charles Hartsell, Nagabhushan Mahadevan, Shreyas Ramakrishna, Abhishek Dubey, Theodore Bapty, Taylor Johnson, Xenofon Koutsoukos, Janos Sztipanovits, and Gabor Karsai. 2019. Model-based design for CPS with learning-enabled components. In *Proceedings of the Workshop on Design Automation for CPS and IoT - DESTION'19*. ACM Press, 1–9. <https://doi.org/10.1145/3313151.3313166>
- [77] Florian Hauer, Tabea Schmidt, Bernd Holzmuller, and Alexander Pretschner. 2019. Did we test all scenarios for automated and autonomous driving systems? In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2950–2955. <https://doi.org/10.1109/ITSC.2019.8917326>
- [78] Peter Henderson, Koustuv Sinha, Nicolas Angelard-Gontier, Nan Rosemary Ke, Genevieve Fried, Ryan Lowe, and Joelle Pineau. 2017. Ethical challenges in data-driven dialogue systems. *arXiv* (2017), 123–129.

- [79] Jens Henriksson, Markus Borg, and Cristofer Englund. 2018. Automotive safety and machine learning. In *Proceedings of the 1st International Workshop on Software Engineering for AI in Autonomous Systems*. ACM, 47–49. <https://doi.org/10.1145/3194085.3194090>
- [80] Charles Hill, Rachel Bellamy, Thomas Erickson, and Margaret Burnett. 2016. Trials and tribulations of developers of intelligent systems: A field study. In *2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, 162–170. <https://doi.org/10.1109/VLHCC.2016.7739680>
- [81] Kenneth Holstein, Jennifer Wortman Vaughan, Hal Daumé, Miroslav Dudík, and Hanna Wallach. 2018. Improving fairness in machine learning systems: What do industry practitioners need? *arXiv* (2018), 1–16.
- [82] 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. <https://doi.org/10.1109/RE.2019.00050>
- [83] Song Huang. 2018. Challenges of testing machine learning applications. *International Journal of Performability Engineering* (2018), 1275–1282. <https://doi.org/10.23940/ijpe.18.06.p18.12751282>
- [84] Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. 2017. Safety verification of deep neural networks. In *Computer Aided Verification*. Springer International Publishing, 3–29. https://doi.org/10.1007/978-3-319-63387-9_1 arXiv:1610.06940.
- [85] Waldemar Hummer, Vinod Muthusamy, Thomas Rausch, Parijat Dube, Kaoutar El Maghraoui, Anupama Murthi, and Punleuk Oum. 2019. ModelOps: Cloud-based lifecycle management for reliable and trusted AI. In *2019 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, 113–120. <https://doi.org/10.1109/IC2E.2019.00025>
- [86] Felix Ingrand. 2019. Recent trends in formal validation and verification of autonomous robots software. In *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE, 321–328. <https://doi.org/10.1109/IRC.2019.00059>
- [87] International Organization For Standardization. 2011. ISO/IEC 25010 - Systems and Software Engineering - Systems and Software Quality Requirements and Evaluation (SQuaRE) - System and Software Quality Models. 25 pages. http://www.iso.org/iso/_catalogue/catalogue_{_}tc/catalogue_{_}detail.htm?csnumber=35733.
- [88] Fuyuki Ishikawa. 2018. Concepts in quality assessment for machine learning - from test data to arguments. In *Conceptual Modeling*. Springer International Publishing, 536–544. https://doi.org/10.1007/978-3-030-00847-5_39
- [89] Fuyuki Ishikawa and Yutaka Matsuno. 2018. Continuous argument engineering: Tackling uncertainty in machine learning based systems. In *Developments in Language Theory*. Springer International Publishing, 14–21. https://doi.org/10.1007/978-3-319-99229-7_2
- [90] Fuyuki Ishikawa and Nobukazu Yoshioka. 2019. How do engineers perceive difficulties in engineering of machine-learning systems? - Questionnaire survey. In *2019 IEEE/ACM Joint 7th International Workshop on Conducting Empirical Studies in Industry (CESI) and 6th International Workshop on Software Engineering Research and Industrial Practice (SER&IP)*. IEEE, 2–9. <https://doi.org/10.1109/CESER-IP.2019.00009>
- [91] Md Johirul Islam, Giang Nguyen, Rangeet Pan, and Hridesh Rajan. 2019. A comprehensive study on deep learning bug characteristics. *arXiv* (2019), 510–520.
- [92] Md Johirul Islam, Hoan Anh Nguyen, Rangeet Pan, and Hridesh Rajan. 2019. What do developers ask about ML libraries? A large-scale study using stack overflow. *arXiv MI* (2019). arXiv:1906.11940.
- [93] Martin Ivarsson and Tony Gorscak. 2010. A method for evaluating rigor and industrial relevance of technology evaluations. *Empirical Software Engineering* 16, 3 (Oct 2010), 365–395. <https://doi.org/10.1007/s10664-010-9146-4>
- [94] Eric Jenn, Alexandre Albore, Franck Mamalet, Grégory Flandin, Christophe Gabreau, Hervé Delseny, Adrien Gaufrisiau, Hugues Bonnin, Lucian Alecu, Jérémie Pirard, Baptiste Lefevre, Jean-Marc Gabriel, Cyril Cappi, Laurent Gardès, Sylvaine Picard, Gilles Dulon, Brice Beltran, Jean-Christophe Banic, Mathieu Damour, Kevin Delmas, and Claire Pagetti. 2020. Identifying challenges to the certification of machine learning for safety critical systems. In *Proceedings of the 10th European Congress on Embedded Real Time Systems (ERTS)*. 10.
- [95] Sophie F. Jentszsch and Nico Hochgeschwender. 2019. Don't forget your roots! Using provenance data for transparent and explainable development of machine learning models. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering Workshop (ASEW)*. IEEE, 37–40. <https://doi.org/10.1109/ASEW.2019.00025>
- [96] Yujie Ji, Xinyang Zhang, Shouling Ji, Xiapu Luo, and Ting Wang. 2018. Model-reuse attacks on deep learning systems. *arXiv* (2018), 349–363.
- [97] Minghua Jia, Xiaodong Wang, Yue Xu, Zhanqi Cui, and Ruilin Xie. 2020. Testing machine learning classifiers based on compositional metamorphic relations. *International Journal of Performability Engineering* 16, 1 (2020), 67. <https://doi.org/10.23940/ijpe.20.01.p8.6777>
- [98] Garazi Juez, Estibaliz Amparan, Ray Lattarulo, Joshue Perez Rastelli, Alejandra Ruiz, and Huascar Espinoza. 2017. Safety assessment of automated vehicle functions by simulation-based fault injection. In *2017 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. IEEE, 214–219. <https://doi.org/10.1109/ICVES.2017.7991928>
- [99] Mary Beth Kery, Marissa Radensky, Mahima Arya, Bonnie E. John, and Brad A. Myers. 2018. The story in the notebook. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 1–11. <https://doi.org/10.1145/3173574.3173748>

- [100] Hourieh Khalajzadeh, Mohamed Abdelrazek, John Grundy, John Hosking, and Qiang He. 2018. A survey of current end-user data analytics tool support. In *2018 IEEE International Congress on Big Data (BigData Congress)*. IEEE, 41–48. <https://doi.org/10.1109/BigDataCongress.2018.00013>
- [101] Foutse Khomh, Bram Adams, Jinghui Cheng, Marios Fokaefs, and Giuliano Antoniol. 2018. Software engineering for machine-learning applications: The road ahead. *IEEE Software* 35, 5 (2018), 81–84.
- [102] Miryung Kim, Thomas Zimmermann, Robert DeLine, and Andrew Begel. 2018. Data scientists in software teams: State of the art and challenges. *IEEE Transactions on Software Engineering* 44, 11 (Nov 2018), 1024–1038. <https://doi.org/10.1109/TSE.2017.2754374>
- [103] Barbara Kitchenham. 2004. Procedures for performing systematic reviews. *Keele, UK, Keele University* 33, 2004 (2004), 1–26.
- [104] Barbara Kitchenham and Stuart Charters. 2007. Guidelines for performing systematic literature reviews in software engineering. Keele University and University of Durham.
- [105] Florian Klueck, Yihao Li, Mihai Nica, Jianbo Tao, and Franz Wotawa. 2018. Using ontologies for test suites generation for automated and autonomous driving functions. In *2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 118–123. <https://doi.org/10.1109/ISSREW.2018.00-20>
- [106] Philip Koopman and Michael Wagner. 2016. Challenges in autonomous vehicle testing and validation. *SAE International Journal of Transportation Safety* 4, 1 (Apr 2016), 15–24. <https://doi.org/10.4271/2016-01-0128>
- [107] Philip Koopman and Michael Wagner. 2018. Toward a framework for highly automated vehicle safety validation. In *SAE Technical Paper Series*. SAE International, 1–13. <https://doi.org/10.4271/2018-01-1071>
- [108] Mark Koren and Mykel J. Kochenderfer. 2019. Efficient autonomy validation in simulation with adaptive stress testing. *arXiv* (2019), 4178–4183.
- [109] Kaan Koseler, Kelsea McGraw, and Matthew Stephan. 2019. Realization of a machine learning domain specific modeling language: A baseball analytics case study. In *Proceedings of the 7th International Conference on Model-Driven Engineering and Software Development*. SciTePress - Science and Technology Publications, 13–24. <https://doi.org/10.5220/0007245800130024>
- [110] Blagovesta Kostova, Seda Gürses, and Alain Wegmann. 2020. On the interplay between requirements, engineering, and artificial intelligence. *CEUR Workshop Proceedings* 2584 (2020).
- [111] Niklas Kühl, Marc Goutier, Robin Hirt, and Gerhard Satzger. 2019. Machine learning in artificial intelligence: Towards a common understanding. In *52nd Hawaii International Conference on System Sciences, HICSS 2019, Grand Wailea, Maui, Hawaii, USA, January 8-11, 2019*, Tung Bui (Ed.). ScholarSpace, 1–10. <http://hdl.handle.net/10125/59960>.
- [112] Marco Kuhrmann, Daniel Méndez Fernández, and Maya Daneva. 2017. On the pragmatic design of literature studies in software engineering: An experience-based guideline. *Empirical Software Engineering* 22, 6 (Jan 2017), 2852–2891. <https://doi.org/10.1007/s10664-016-9492-y>
- [113] Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. 2015. Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*. ACM, 126–137. <https://doi.org/10.1145/2678025.2701399>
- [114] Abhishek Kumar, Tristan Braud, Sasu Tarkoma, and Pan Hui. 2020. Trustworthy AI in the age of pervasive computing and big data. *arXiv* (2020). [arXiv:2002.05657](https://doi.org/10.4236/ojs.2020010010001).
- [115] Fumihiro Kumeno. 2019. Software engineering challenges for machine learning applications: A literature review. *Intelligent Decision Technologies* 13, 4 (2019), 463–476.
- [116] Hiroshi Kuwajima and Fuyuki Ishikawa. 2019. Adapting SQuaRE for quality assessment of artificial intelligence systems. In *2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 13–18. <https://doi.org/10.1109/ISSREW.2019.00035> [arXiv:1908.02134](https://doi.org/10.4236/ojs.20190906134).
- [117] Hiroshi Kuwajima, Hirotoshi Yasuoka, and Toshihiro Nakae. 2019. Open Problems in Engineering Machine Learning Systems and the Quality Model. *arXiv* (2019). [arXiv:1904.00001v1](https://doi.org/10.4236/ojs.20190906134).
- [118] Hiroshi Kuwajima, Hirotoshi Yasuoka, and Toshihiro Nakae. 2020. Engineering problems in machine learning systems. *Machine Learning* 109, 5 (Apr 2020), 1103–1126. <https://doi.org/10.1007/s10994-020-05872-w> [arXiv:1904.00001](https://doi.org/10.4236/ojs.20190906134).
- [119] Christian Kästner and Eunsuk Kang. 2020. Teaching software engineering for AI-enabled systems. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training*. ACM, 45–48. <https://doi.org/10.1145/3377814.3381714> [arXiv:2001.06691](https://doi.org/10.4236/ojs.20190906134).
- [120] Shuyue Lan, Chao Huang, Zhilu Wang, Hengyi Liang, Wenhao Su, and Qi Zhu. 2018. Design automation for intelligent automotive systems. In *2018 IEEE International Test Conference (ITC)*. IEEE, 1–10. <https://doi.org/10.1109/TEST.2018.8624723>
- [121] Francesco Leofante, Luca Pulina, and Armando Tacchella. 2016. Learning with safety requirements: State of the art and open questions. *CEUR Workshop Proceedings* 1745 (2016), 11–25.
- [122] Maurizio Leotta, Dario Olianas, Filippo Ricca, and Nicoletta Noceti. 2019. How do implementation bugs affect the results of machine learning algorithms? In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*. ACM, 1304–1313. <https://doi.org/10.1145/3297280.3297411>

- [123] David C. Liu, Stephanie Rogers, Raymond Shiao, Dmitry Kislyuk, Kevin C. Ma, Zhigang Zhong, Jenny Liu, and Yushi Jing. 2017. Related pins at Pinterest. In *Proceedings of the 26th International Conference on World Wide Web Companion - WWW'17 Companion*. ACM Press, 583–592. <https://doi.org/10.1145/3041021.3054202>
- [124] 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).
- [125] Lucy Ellen Lwakatare, Aiswarya Raj, Jan Bosch, Helena Holmström Olsson, and Ivica Crnkovic. 2019. A taxonomy of software engineering challenges for machine learning systems: An empirical investigation. In *Lecture Notes in Business Information Processing*. Springer International Publishing, 227–243. https://doi.org/10.1007/978-3-030-19034-7_14
- [126] 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.
- [127] Lei Ma, Fuyuan Zhang, Minhui Xue, Bo Li, Yang Liu, Jianjun Zhao, and Yadong Wang. 2018. Combinatorial testing for deep learning systems. *arXiv* (2018), 614–618. arXiv:[1806.07723](https://doi.org/10.4236/arxiv.180607723).
- [128] Shiqing Ma, Yingqi Liu, Wen-Chuan Lee, Xiangyu Zhang, and Ananth Grama. 2018. MODE: Automated neural network model debugging via state differential analysis and input selection. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, 175–186. <https://doi.org/10.1145/3236024.3236082>
- [129] Fumio Machida. 2019. N-version machine learning models for safety critical systems. In *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, 48–51. <https://doi.org/10.1109/DSN-W.2019.00017>
- [130] Fumio Machida. 2019. On the diversity of machine learning models for system reliability. In *2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 276–285. <https://doi.org/10.1109/PRDC47002.2019.00058>
- [131] Rupak Majumdar, Aman Mathur, Marcus Pirron, Laura Stegner, and Damien Zufferey. 2019. Paracosm: A language and tool for testing autonomous driving systems. *arXiv* (2019). arXiv:[1902.01084](https://doi.org/10.4236/arxiv.1902.01084).
- [132] Piergiuseppe Mallozzi, Patrizio Pelliccione, and Claudio Menghi. 2018. Keeping intelligence under control. In *Proceedings of the 1st International Workshop on Software Engineering for Cognitive Services*. ACM, 37–40. <https://doi.org/10.1145/3195555.3195558>
- [133] Silverio Martínez-Fernández, Xavier Franch, Andreas Jedlitschka, Marc Oriol, and Adam Trendowicz. 2020. Research directions for developing and operating artificial intelligence models in trustworthy autonomous systems. *arXiv* (2020). arXiv:[2003.05434](https://doi.org/10.4236/arxiv.200305434).
- [134] 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.
- [135] David Issa Mattos, Jan Bosch, and Helena Holmström Olsson. 2019. Leveraging business transformation with machine learning experiments. In *Lecture Notes in Business Information Processing*. Springer International Publishing, 183–191. https://doi.org/10.1007/978-3-030-33742-1_15
- [136] John McDermid, Yan Jia, and Ibrahim Habli. 2019. Towards a framework for safety assurance of autonomous systems. *CEUR Workshop Proceedings* 2419 (2019).
- [137] Meenu Mary John, Helena Holmström Olsson, and Jan Bosch. [n.d.]. Architecting AI deployment: A systematic review of state-of-the-art and state-of-practice literature.
- [138] Tim Menzies. 2020. The five laws of SE for AI. *IEEE Software* 37, 1 (Jan 2020), 81–85. <https://doi.org/10.1109/MS.2019.2954841>
- [139] Caroline Bianca Santos Tancredi Molina, Jorge Rady de Almeida, Lucio F. Vismari, Rodrigo Ignacio R. Gonzalez, Jamil K. Naufal, and Joao Batista Camargo. 2017. Assuring fully autonomous vehicles safety by design: The autonomous vehicle control (AVC) module strategy. In *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, 16–21. <https://doi.org/10.1109/DSN-W.2017.14>
- [140] Mohammed Moreb, Tareq Abed Mohammed, Oguz Bayat, and Oguz Ata. 2020. Corrections to “A novel software engineering approach toward using machine learning for improving the efficiency of health systems”. *IEEE Access* 8 (2020), 136459–136459. <https://doi.org/10.1109/ACCESS.2020.2986259>
- [141] 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 (Jul 2020), 106294. <https://doi.org/10.1016/j.infsof.2020.106294>
- [142] Aiswarya Munappy, Jan Bosch, Helena Holmstrom Olsson, Anders Arpteg, and Bjorn Brinne. 2019. Data management challenges for deep learning. In *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 140–147. <https://doi.org/10.1109/SEAA.2019.00030>

- [143] Shin Nakajima. 2018. [Invited] Quality assurance of machine learning software. In *2018 IEEE 7th Global Conference on Consumer Electronics (GCCE)*. IEEE, 143–144. <https://doi.org/10.1109/GCCE.2018.8574766>
- [144] Shin Nakajima. 2019. Dataset diversity for metamorphic testing of machine learning software. In *Structured Object-Oriented Formal Language and Method*, Zhenhua Duan, Shaoying Liu, Cong Tian, and Fumiko Nagoya (Eds.). Springer International Publishing, Cham, 21–38. https://doi.org/10.1007/978-3-030-13651-2_2
- [145] Shin Nakajima. 2019. Quality evaluation assurance levels for deep neural networks software. In *2019 International Conference on Technologies and Applications of Artificial Intelligence (TAAI)*. IEEE. <https://doi.org/10.1109/TAAI48200.2019.8959916>
- [146] Soroosh Nalchigar, Eric Yu, Yazan Obeidi, Sebastian Carabajales, John Green, and Allen Chan. 2019. Solution patterns for machine learning. In *Advanced Information Systems Engineering*. Springer International Publishing, 627–642. https://doi.org/10.1007/978-3-030-21290-2_39
- [147] 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).
- [148] Peter Naur, Brian Randell, Friedrich Ludwig Bauer, and NATO Science Committee. (Eds.). 1969. *Software Engineering: Report on a Conference Sponsored by the NATO Science Committee, Garmisch, Germany, 7th to 11th October 1968*. Scientific Affairs Division, NATO.
- [149] Yasuharu Nishi, Satoshi Masuda, Hideto Ogawa, and Keiji Uetsuki. 2018. A test architecture for machine learning product. In *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 273–278. <https://doi.org/10.1109/ICSTW.2018.00060>
- [150] Besmira Nushi, Ece Kamar, and Eric Horvitz. 2018. Towards accountable AI: Hybrid human-machine analyses for characterizing system failure. *arXiv Hcomp* (2018), 126–135. arXiv:1809.07424.
- [151] Augustus Odena and Ian Goodfellow. 2018. Tensorfuzz: Debugging neural networks with coverage-guided fuzzing. *arXiv* (2018).
- [152] Carlos E. Otero and Adrian Peter. 2015. Research directions for engineering big data analytics software. *IEEE Intelligent Systems* 30, 1 (Jan 2015), 13–19. <https://doi.org/10.1109/MIS.2014.76>
- [153] Ipek Ozkaya. 2020. What is really different in engineering AI-enabled systems? *IEEE Software* 37, 4 (Jul 2020), 3–6. <https://doi.org/10.1109/MS.2020.2993662>
- [154] D. Partridge and Y. Wilks. 1987. Does AI have a methodology which is different from software engineering? *Artificial Intelligence Review* 1, 2 (1987), 111–120. <https://doi.org/10.1007/BF00130012>
- [155] Kayur Patel. 2010. Lowering the barrier to applying machine learning. In *Adjunct Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology - UIST'10*. ACM Press, 355–358. <https://doi.org/10.1145/1866218.1866222>
- [156] Gabriel Pedroza and Adedjouma Morayo. 2019. Safe-by-design development method for artificial intelligent based systems. In *Proceedings of the 31st International Conference on Software Engineering and Knowledge Engineering*. KSI Research Inc. and Knowledge Systems Institute Graduate School, 391–397. <https://doi.org/10.18293/SEKE2019-094>
- [157] Mirko Perkusich, Lenardo Chaves e Silva, Alexandre Costa, Felipe Ramos, Renata Saraiva, Arthur Freire, Ednaldo Dilorenzo, Emanuel Dantas, Danilo Santos, Kyller Gorgônio, Hyggo Almeida, and Angelo Perkusich. 2020. Intelligent software engineering in the context of agile software development: A systematic literature review. *Information and Software Technology* 119 (2020), 106241. <https://doi.org/10.1016/j.infsof.2019.106241>
- [158] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. 2008. Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*. BCS Learning & Development, 68–77. <https://doi.org/10.14236/ewic/EASE2008.8>
- [159] Kai Petersen, Sairam Vakkalanka, and Ludwik Kuzniarz. 2015. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology* 64 (Aug 2015), 1–18. <https://doi.org/10.1016/j.infsof.2015.03.007>
- [160] Luca Pulina and Armando Tacchella. 2010. An abstraction-refinement approach to verification of artificial neural networks. *CEUR Workshop Proceedings* 616 (2010), 243–257.
- [161] Mona Rahimi, Jin L. C. Guo, Sahar Kokaly, and Marsha Chechik. 2019. Toward requirements specification for machine-learned components. In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*. IEEE, 241–244. <https://doi.org/10.1109/REW.2019.00049>
- [162] Saidur Rahman, Emilio River, Foutse Khomh, Yann Gal Guhneuc, and Bernd Lehnert. 2019. Machine learning software engineering in practice: An industrial case study. *arXiv* (2019), 1–21. arXiv:1906.07154.
- [163] Inioluwa Deborah Raji, Andrew Smart, Rebecca N. White, Margaret Mitchell, Timnit Gebru, Ben Hutchinson, Jamila Smith-Loud, Daniel Theron, and Parker Barnes. 2020. Closing the AI accountability gap. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. ACM, 33–44. <https://doi.org/10.1145/3351095.3372873> arXiv:2001.00973.

- [164] Paul Ralph, Nauman bin Ali, Sebastian Baltes, Domenico Bianculli, Jessica Diaz, Yvonne Dittrich, Neil Ernst, Michael Felderer, Robert Feldt, Antonio Filieri, Breno Bernard Nicolau de França, Carlo Alberto Furia, Greg Gay, Nicolas Gold, Daniel Graziotin, Pinjia He, Rashina Hoda, Natalia Juristo, Barbara Kitchenham, Valentina Lenarduzzi, Jorge Martínez, Jorge Melegati, Daniel Mendez, Tim Menzies, Jefferson Moller, Dietmar Pfahl, Romain Robbes, Daniel Russo, Nytti Saarimäki, Federica Sarro, Davide Taibi, Janet Siegmund, Diomidis Spinellis, Miroslaw Staron, Klaas Stol, Margaret-Anne Storey, Davide Taibi, Damian Tamburri, Marco Torchiano, Christoph Treude, Burak Turhan, Xiaofeng Wang, and Sira Vegas. 2021. Empirical Standards for Software Engineering Research. arXiv:2010.03525 [cs.SE].
- [165] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. “Why should I trust you?” Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1135–1144. <https://doi.org/10.1145/2939672.2939778> arXiv:1602.04938.
- [166] 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, 6 (2020), 5193–5254. <https://doi.org/10.1007/s10664-020-09881-0>
- [167] R. A. Rill and A. Lörincz. 2019. Cognitive modeling approach for dealing with challenges in cyber-physical systems. *Studia Universitatis Babeș-Bolyai Informatica* 64, 1 (Jun 2019), 51–66. <https://doi.org/10.24193/subbi.2019.1.05>
- [168] Abu Hasnat Mohammad Rubaiyat, Yongming Qin, and Homa Alemezadeh. 2018. Experimental resilience assessment of an open-source driving agent. In *2018 IEEE 23rd Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 54–63. <https://doi.org/10.1109/PRDC.2018.00016> arXiv:1807.06172.
- [169] Stuart J. Russell and Peter Norvig. 2021. *Artificial Intelligence: A Modern Approach* (Fourth edition). Pearson, Hoboken.
- [170] Rick Salay and Krzysztof Czarnecki. 2018. Using machine learning safely in automotive software: An assessment and adaption of software process requirements in ISO 26262. *arXiv* (2018). arXiv:1808.01614.
- [171] Rick Salay and Krzysztof Czarnecki. 2019. Improving ML safety with partial specifications. In *Lecture Notes in Computer Science*. Springer International Publishing, 288–300. https://doi.org/10.1007/978-3-030-26250-1_23
- [172] Rick Salay, Rodrigo Queiroz, and Krzysztof Czarnecki. 2017. An analysis of ISO 26262: Using machine learning safely in automotive software. *arXiv* (2017). arXiv:1709.02435.
- [173] P. Santhanam, Eitan Farchi, and Victor Pankratius. 2019. Engineering reliable deep learning systems. *arXiv* 3 (2019), 1–8. arXiv:1910.12582.
- [174] Prakash Sarathy, Sanjoy Baruah, Stephen Cook, and Marilyn Wolf. 2019. Realizing the promise of artificial intelligence for unmanned aircraft systems through behavior bounded assurance. In *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*. IEEE. <https://doi.org/10.1109/DASC43569.2019.9081649>
- [175] Naoto Sato, Hironobu Kuruma, Masanori Kaneko, Yuichiro Nakagawa, Hideto Ogawa, Thai Son Hoang, and Michael Butler. 2018. DeepSaucer: Unified environment for verifying deep neural networks. *arXiv* (2018). arXiv:1811.03752.
- [176] William Saunders, Andreas Stuhlmüller, Girish Sastry, and Owain Evans. 2018. Trial without error: Towards safe reinforcement learning via human intervention. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS* 3 (2018), 2067–2069. arXiv:1707.05173.
- [177] Sebastian Schelter, Felix Biessmann, Tim Januschowski, David Salinas, Stephan Seufert, and Gyuri Szarvas. 2018. On challenges in machine learning model management. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering* (2018), 5–13. <http://sites.computer.org/debull/A18dec/p5.pdf>.
- [178] Johann Schleier-Smith. 2015. An architecture for agile machine learning in real-time applications. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2059–2068. <https://doi.org/10.1145/2783258.2788628>
- [179] D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean François Crespo, and Dan Dennison. 2015. Hidden technical debt in machine learning systems. *Advances in Neural Information Processing Systems* 2015-Jan (2015), 2503–2511.
- [180] 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.
- [181] Alex Serban and Joost Visser. 2021. An empirical study of software architecture for machine learning. *arXiv preprint arXiv:2105.12422* (2021).
- [182] Alexandru Constantin Serban. 2019. Designing safety critical software systems to manage inherent uncertainty. In *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*. IEEE, 246–249. <https://doi.org/10.1109/ICSA-C.2019.00051>
- [183] Sanjit A. Seshia, Dorsa Sadigh, and S. Shankar Sastry. 2016. Towards verified artificial intelligence. *arXiv* (2016), 1–18. arXiv:1606.08514 <http://arxiv.org/abs/1606.08514>.
- [184] Sina Shafaei, Stefan Kugele, Mohd Hafeez Osman, and Alois Knoll. 2018. Uncertainty in machine learning: A safety perspective on autonomous driving. In *Developments in Language Theory*. Springer International Publishing, 458–464. https://doi.org/10.1007/978-3-319-99229-7_39

- [185] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. 2017. On a formal model of safe and scalable self-driving cars. *arXiv* (2017), 1–37. arXiv:[1708.06374](https://arxiv.org/abs/1708.06374).
- [186] Raymond Sheh and Isaac Monteaeth. 2018. Defining explainable AI for requirements analysis. *KI - Künstliche Intelligenz* 32, 4 (Oct 2018), 261–266. <https://doi.org/10.1007/s13218-018-0559-3>
- [187] Patrice Y. Simard, Saleema Amershi, David M. Chickering, Alicia Edelman Pelton, Soroush Ghorashi, Christopher Meek, Gonzalo Ramos, Jina Suh, Johan Verwey, Mo Wang, and John Wernsing. 2017. Machine teaching a new paradigm for building machine learning systems. *arXiv* (2017). arXiv:[1707.06742](https://arxiv.org/abs/1707.06742).
- [188] Helge Spieker and Arnaud Gotlieb. 2019. Towards testing of deep learning systems with training set reduction. *arXiv* 2 (2019). arXiv:[1901.04169](https://arxiv.org/abs/1901.04169).
- [189] Siwakorn Srisakaokul, Yuhao Zhang, Zexuan Zhong, Wei Yang, Tao Xie, and Bo Li. 2018. MuDef: Multi-model-based defense against adversarial examples for neural networks. *arXiv* (2018). arXiv:[1809.00065](https://arxiv.org/abs/1809.00065).
- [190] Xiaobing Sun, Tianchi Zhou, Gengjie Li, Jiajun Hu, Hui Yang, and Bin Li. 2017. An empirical study on real bugs for machine learning programs. In *2017 24th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 348–357. <https://doi.org/10.1109/APSEC.2017.41>
- [191] Youcheng Sun, Xiaowei Huang, Daniel Kroening, James Sharp, Matthew Hill, and Rob Ashmore. 2019. DeepConcolic: Testing and debugging deep neural networks. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. IEEE, 111–114. <https://doi.org/10.1109/ICSE-Companion.2019.00051>
- [192] Youcheng Sun, Xiaowei Huang, Daniel Kroening, James Sharp, Matthew Hill, and Rob Ashmore. 2019. Structural test coverage criteria for deep neural networks. *ACM Transactions on Embedded Computing Systems* 18, 5s (Oct 2019), 1–23. <https://doi.org/10.1145/3358233>
- [193] Youcheng Sun, Min Wu, Wenjie Ruan, Xiaowei Huang, Marta Kwiatkowska, and Daniel Kroening. 2018. Concolic testing for deep neural networks. *arXiv* (2018), 109–119.
- [194] Youcheng Sun, Yifan Zhou, Simon Maskell, James Sharp, and Xiaowei Huang. 2020. Reliability validation of learning enabled vehicle tracking. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 9390–9396. <https://doi.org/10.1109/ICRA40945.2020.9196932>. arXiv:[2002.02424](https://arxiv.org/abs/2002.02424)
- [195] Ferdinand Thung, Shaowei Wang, David Lo, and Lingxiao Jiang. 2012. An empirical study of bugs in machine learning systems. In *2012 IEEE 23rd International Symposium on Software Reliability Engineering*. IEEE, 271–280. <https://doi.org/10.1109/ISSRE.2012.22>
- [196] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. DeepTest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th International Conference on Software Engineering*. ACM, 303–314. <https://doi.org/10.1145/3180155.3180220>
- [197] Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. 2016. Stealing machine learning models via prediction APIs. In *25th USENIX Security Symposium (USENIX Security 16)*. <https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/tramer>.
- [198] Cumhur Erkan Tuncali, Georgios Fainekos, Hisahiro Ito, and James Kapinski. 2018. Sim-ATAV. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control (Part of CPS Week)*. ACM, 283–284. <https://doi.org/10.1145/3178126.3187004>
- [199] Cumhur Erkan Tuncali, Georgios Fainekos, Danil Prokhorov, Hisahiro Ito, and James Kapinski. 2020. Requirements-driven test generation for autonomous vehicles with machine learning components. *IEEE Transactions on Intelligent Vehicles* 5, 2 (Jun 2020), 265–280. <https://doi.org/10.1109/TIV.2019.2955903>
- [200] Sakshi Udeshi, Pryanshu Arora, and Sudipta Chattopadhyay. 2018. Automated directed fairness testing. *arXiv* (2018), 98–108.
- [201] Tom van der Weide, Dimitris Papadopoulos, Oleg Smirnov, Michal Zielinski, and Tim van Kasteren. 2017. Versioning for end-to-end machine learning pipelines. In *Proceedings of the 1st Workshop on Data Management for End-to-End Machine Learning*. ACM. <https://doi.org/10.1145/3076246.3076248>
- [202] Kush R. Varshney. 2016. Engineering safety in machine learning. In *2016 Information Theory and Applications Workshop (ITA)*. IEEE. <https://doi.org/10.1109/ITA.2016.7888195>. arXiv:[1601.04126](https://arxiv.org/abs/1601.04126).
- [203] Kush R. Varshney and Homa Alemzadeh. 2017. On the safety of machine learning: Cyber-physical systems, decision sciences, and data products. *Big Data* 5, 3 (Sep 2017), 246–255. <https://doi.org/10.1089/big.2016.0051>. arXiv:[1610.01256](https://arxiv.org/abs/1610.01256).
- [204] Marisa Vasconcelos, Heloisa Candello, Claudio Pinhanez, and Thiago dos Santos. 2017. Bottester. In *Proceedings of the XVI Brazilian Symposium on Human Factors in Computing Systems*. ACM, 1–4. <https://doi.org/10.1145/3160504.3160584>
- [205] Andreas Vogelsang and Markus Borg. 2019. Requirements engineering for machine learning: Perspectives from data scientists. In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*. IEEE, 245–251. <https://doi.org/10.1109/REW.2019.00050>. arXiv:[1908.04674](https://arxiv.org/abs/1908.04674).
- [206] Zhiyuan Wan, Xin Xia, David Lo, and Gail C. Murphy. 2020. How does machine learning change software development practices? *IEEE Transactions on Software Engineering* (2020), 1–15. <https://doi.org/10.1109/TSE.2019.2937083>

- [207] Jingyi Wang, Jun Sun, Peixin Zhang, and Xinyu Wang. 2018. Detecting adversarial samples for deep neural networks through mutation testing. *arXiv* (2018), 1–10. arXiv:1805.05010.
- [208] 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).
- [209] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. 2018. Efficient formal safety analysis of neural networks. *arXiv NeurIPS* (2018).
- [210] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. 2018. Formal security analysis of neural networks using symbolic intervals. *arXiv* (2018).
- [211] 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, 49–54.
- [212] 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 - EASE'14*. ACM Press, New York, NY, USA, Article 38. <https://doi.org/10.1145/2601248.2601268>
- [213] Claes Wohlin, Per Runeson, Paulo Anselmo da Mota Silveira Neto, Emelie Engström, Ivan do Carmo Machado, and Eduardo Santana de Almeida. 2013. On the reliability of mapping studies in software engineering. *Journal of Systems and Software* 86, 10 (Oct 2013), 2594–2610. <https://doi.org/10.1016/j.jss.2013.04.076>
- [214] Christine T. Wolf and Drew Paine. 2020. Sensemaking practices in the everyday work of AI/ML software engineering. In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*. ACM, 86–92. <https://doi.org/10.1145/3387940.3391496>
- [215] Christian Wolschke, Thomas Kuhn, Dieter Rombach, and Peter Liggesmeyer. 2017. Observation based creation of minimal test suites for autonomous vehicles. In *2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 294–301. <https://doi.org/10.1109/ISSREW.2017.46>
- [216] W. Eric Wong, Nikolaos Mittas, Elvira Maria Arvanitou, and Yihao Li. 2021. A bibliometric assessment of software engineering themes, scholars and institutions (2013–2020). *Journal of Systems and Software* 180 (2021), 111029. <https://doi.org/10.1016/j.jss.2021.111029>
- [217] Weibin Wu, Hui Xu, Sanqiang Zhong, Michael R. Lyu, and Irwin King. 2019. Deep validation: Toward detecting real-world corner cases for deep neural networks. In *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 125–137. <https://doi.org/10.1109/DSN.2019.00026>
- [218] Tao Xie. 2018. Intelligent software engineering: Synergy between AI and software engineering. In *Dependable Software Engineering. Theories, Tools, and Applications*, Xinyu Feng, Markus Müller-Olm, and Zijiang Yang (Eds.). Springer International Publishing, Cham, 3–7.
- [219] Xiaofei Xie, Lei Ma, Felix Juefei-Xu, Minhui Xue, Hongxu Chen, Yang Liu, Jianjun Zhao, Bo Li, Jianxiong Yin, and Simon See. 2019. DeepHunter: A coverage-guided fuzz testing framework for deep neural networks. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ACM, 158–168. <https://doi.org/10.1145/3293882.3330579>
- [220] Xiaofei Xie, Lei Ma, Haijun Wang, Yuekang Li, Yang Liu, and Xiaohong Li. 2019. DiffChaser: Detecting disagreements for deep neural networks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, 5772–5778. <https://doi.org/10.24963/ijcai.2019/800>
- [221] Shakiba Yaghoubi and Georgios Fainekos. 2018. Gray-box adversarial testing for control systems with machine learning component. *arXiv* (2018), 179–184.
- [222] Qian Yang. 2017. The Role of Design in Creating Machine-Learning-enhanced User Experience. *AAAI Spring Symposium - Technical Report SS-17-01* - (2017), 406–411.
- [223] Wei Yang and Tao Xie. 2018. Telemade: A testing framework for learning-based malware detection systems. *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence* (2018), 400–403.
- [224] Zhuolin Yang, Zhikuan Zhao, Hengzhi Pei, Boxin Wang, Bojan Karlas, Ji Liu, Heng Guo, Bo Li, and Ce Zhang. 2020. End-to-end robustness for sensing-reasoning machine learning pipelines. *arXiv* (2020), 1–43. arXiv:2003.00120.
- [225] Haruki Yokoyama. 2019. Machine learning system architectural pattern for improving operational stability. In *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*. IEEE, 267–274. <https://doi.org/10.1109/ICSA-C.2019.00055>
- [226] Jie M. Zhang, Mark Harman, Lei Ma, and Yang Liu. 2020. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering* (2020).
- [227] Tianyi Zhang, Cuiyun Gao, Lei Ma, Michael Lyu, and Miryung Kim. 2019. An empirical study of common challenges in developing deep learning applications. In *2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 104–115. <https://doi.org/10.1109/ISSRE.2019.00020>

- [228] Xufan Zhang, Yilin Yang, Yang Feng, and Zhenyu Chen. 2019. Software engineering practice in the development of deep learning applications. *arXiv* (2019). arXiv:[1910.03156](https://arxiv.org/abs/1910.03156).
- [229] Yuhao Zhang, Yifan Chen, Shing-Chi Cheung, Yingfei Xiong, and Lu Zhang. 2018. An empirical study on TensorFlow program bugs. In *Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ACM, 129–140. <https://doi.org/10.1145/3213846.3213866>
- [230] Shuai Zhao, Manoop Talasila, Guy Jacobson, Cristian Borcea, Syed Anwar Aftab, and John F. Murray. 2018. Packaging and sharing machine learning models via the Acumos AI open platform. *arXiv* (2018).
- [231] Xinghan Zhao and Xiangfei Gao. 2018. An AI software test method based on scene deductive approach. In *2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*. IEEE, 14–20. <https://doi.org/10.1109/QRS-C.2018.00017>
- [232] Wujie Zheng, Wenyu Wang, Dian Liu, Changrong Zhang, Qinsong Zeng, Yuetang Deng, Wei Yang, Pinjia He, and Tao Xie. 2019. Testing untestable neural machine translation: An industrial case. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. IEEE, 314–315. <https://doi.org/10.1109/ICSE-Companion.2019.00131> arXiv:[1807.02340](https://arxiv.org/abs/1807.02340).
- [233] Husheng Zhou, Wei Li, Zelun Kong, Junfeng Guo, Yuqun Zhang, Bei Yu, Lingming Zhang, and Cong Liu. 2020. DeepBillboard: Systematic physical-world testing of autonomous driving systems. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ACM, 347–358. <https://doi.org/10.1145/3377811.3380422> arXiv:[1812.10812](https://arxiv.org/abs/1812.10812).

Received May 2021; revised July 2021; accepted August 2021