Quality Assurance for Artificial Intelligence: A Study of Industrial Key Concerns, Challenges and Best Practices

Quality Assurance (QA) aims to prevent mistakes and defects in manufactured products and avoid problems when delivering products or services to customers. QA for AI systems, however, poses particular challenges, given their data-driven and non-deterministic nature, the large scale of deployment as well as more complex architectures and algorithms. While there is growing empirical evidence about practices of machine learning in industrial contexts, little is known about the challenges and best practices of quality assurance for AI systems (QA4AI). In this paper, we report on a mixed-method study of AI and its QA in industry practice from various countries and companies. Through interviews with fifteen industry practitioners and a validation survey with 50 practitioner responses, we studied the importance as well as challenges and best practices in ensuring the main aspects of quality for AI systems reported in the literature such as *correctness*, *model relevance* (the alignment between an AI model's capabilities and its target data in the application environment), fairness, interpretability and others. Our findings suggest *correctness* as the most important property, followed by *model relevance*, *efficiency* and *deployability*. In contrast, *transferability* (applying knowledge learned in one context to another context or task), *security* and *fairness* are not paid much attention by practitioners compared to other properties. When analyzing practitioner challenges in ensuring such quality properties, interviewees highlight the trade-off among latency, cost and accuracy as the big challenge for *efficiency* (latency and cost are parts of *efficiency* concern). Solutions such as model compression, use of CPU-only server, and efficient model pipeline are proposed. Finally, our findings provide 21 QA4AI practices across each stage of AI development.

CCS Concepts: • **Software and its engineering → Empirical software validation**; • **Computing methodologies → Artificial intelligence**.

Additional Key Words and Phrases: AI Engineering, Machine Learning, Quality Assurance, Interview, Survey

ACM Reference Format:
. 2018. Quality Assurance for Artificial Intelligence: A Study of Industrial Key Concerns, Challenges and Best Practices. 1, 1 (February 2018), 41 pages. https://doi.org/10.1145/xxxxx.xxxxxx

1 INTRODUCTION

Recent decades have witnessed the rapid growth of Artificial Intelligence (AI) technologies in a vast spectrum of critical domains, including finance [22, 27], image processing [47, 74], software development [105], etc. However, same as many other software systems, AI systems are prone to bugs, failures, and other quality issues that can lead to severe consequences. For example, according to news in June 2022, nearly 400 car crashes in 11 months involved automated technologies. [100] Even ChatGPT, a popular AI chatbot, is exploitable in certain ways [79, 109]. As a result, understanding how to ensure the quality of the AI system is a vital task that can have a significant impact on the stakeholders.

Author's address:

XXXX-XXXX/2018/2-ART $15.00
https://doi.org/10.1145/xxxxx.xxxxxx

As defined by the International Organization for Standardization (ISO) 9000, *Quality Assurance* (QA) is "part of quality management focused on providing confidence that quality requirements will be fulfilled." [13] Many QA techniques, e.g., bug detection methods [49, 68, 92, 98, 121, 139] and CI/CD tools [99] have been developed to assure the quality of traditional software that encode their behaviors using data and control logics implement by traditional programming languages. As the QA techniques for traditional software may not be directly applicable to AI systems, researchers have proposed a series of properties to evaluate various quality aspects of AI systems (e.g., fairness [21, 101, 127], robustness [38, 46, 72], correctness [55, 96, 128]).

However, as pointed out by Song et al. [113], academia-based research papers about QA for AI systems cannot be easily practiced by industry practitioners. Additionally, AI systems in the industry are usually larger and more complex than the ones evaluated in research papers. The standards and regulations they need to meet are usually many-sided and domain-specific. For example, AI systems for finance need to meet the requirements of the *Basel Accords* [12], i.e., a set of agreements on banking regulations. Data used by AI systems in the domain that involves private personal information need to comply with the privacy regulations of operating countries, such as *PDPA* [7] in Singapore and *GDPR* [14] in the European Union.

These considerations underscore the importance of gaining a thorough understanding of Quality Assurance for AI systems (QA4AI) from the perspective of industry practitioners. While there exists research exploring the challenges and practices of machine learning in industry context [94, 108], to the best of our knowledge, there exist some research gaps that need to be addressed. First, it is unclear how practitioners perceive the importance of each QA4AI property. Understanding such perception can help prioritize their efforts in QA for AI systems. Second, it lacks understanding of the challenges, corresponding solutions and best practices of different QA properties. Some studies focus on the challenges encountered in the development of AI systems [28, 89, 94], but they do not explicitly focus on the challenges of QA for AI systems, or only focus on one particular QA4AI property, e.g., fairness [39]. Similarly, current studies for AI best practices also do not explicitly focus on QA for AI systems [80, 108].

To fill this research gap, we conduct a mixed methods study through a combination of interviews with 15 AI practitioners, and surveys with 50 additional practitioners to validate our findings. We examine 9 QA4AI properties proposed by Zhang et al. [137] (*correctness*, *model relevance*, *robustness*, *security*, *efficiency*, *fairness*, *interpretability* and *privacy*) and add 3 additional properties (*deployability*, *transferability* and *scalability*) inspired from the literature, which we will explain in Section 2. We then interview practitioners about the awareness and current practices of 11 QA4AI properties, as well as detailed questions for each step in the AI product lifecycle. For each QA4AI property, we analyze interviewees' ratings about its importance and the underlying reasons in RQ1, the major challenges and possible solutions in RQ2. Furthermore, we also extract 21 best practices for QA4AI from the interview study, which is further validated in a validation survey with 50 additional practitioners in RQ3.

To the best of our knowledge, this is the first study that provides a thorough understanding of QA for AI systems from the perspective of industry practitioners through focus, challenge and best practice analysis. This study provides a detailed understanding of QA4AI from the industry practitioners' perspective.

The contributions of this paper are as follows:

- **Investigation of the Significance of QA4AI Properties:** This study investigates the importance score ranked by industry participants regarding 11 QA4AI properties. Findings are extracted from the importance distribution of each property.

- **Identification & Analysis of QA4AI Challenges and Solutions:** This study identifies the challenges of each QA4AI property extracted from the interview study, and possible solutions along with each of the challenges are also analyzed.
- **Identification & Validation of QA4AI Best Practices:** This study extracts 21 best practices for QA4AI from the interview study, which is then further validated in the validation survey, with 10 out of 21 best practices being well-supported (average rating greater than 4, i.e., 'accept') by the practitioners and 8 out of 21 practices are marginally agreed (average rating greater than 3.5, i.e., middle of 'agree' and 'neutral') by the participants.
- **Recommendations to Practitioners:** After the analysis of the interview and validation survey, we provide suggestions to practitioners on how to improve their QA4AI process about some concepts that frequently occur in the interview result, such as "*Dos and don'ts for real-time AI systems*" and "*Effective Utilization of Open Source Resources*".

The structure of this paper is organized as follows: Section 2 provides an introduction to the background of artificial intelligence (AI), delving into aspects of AI, AI quality and QA4AI. In Section 3, we outline the methodology of our interview study and validation survey. Section 4 is dedicated to the derivation of three research questions (RQs), focusing on 11 properties of QA for AI (QA4AI) and 21 best practices. Following this, Section 5 discusses recommendations for practitioners and proposes potential threats to validity. Section 6 presents a review of related works. Finally, Section 7 concludes the paper, encapsulating the key findings and contributions, and proposes future work.

2 BACKGROUND

This section presents the background of this paper, including the description of AI, AI quality, and quality assurance for AI.

2.1 AI Quality

Quality is critical for the success of AI systems. Many studies have explored AI quality issues from different perspectives. Heyn et al. [61] pointed out four requirement engineering challenge areas for AI-intense systems development that may influence the quality of AI systems: contextual definitions and requirements, data attributes and requirements, performance definition and monitoring, and human factors. Whang et al. [123] conducted a study on data quality challenges in AI systems, they cover data validation, cleaning and integration techniques and identify the challenges in each of these areas, for example, small, dirty, biased, poisoned data, and identify many effective solutions to address these challenges. Ying [132] systematically analyzed one of the most common quality issues during model training: overfitting, with a deep analysis of its cause and effective solutions.

2.2 Quality Assurance Properties for AI systems

QA4AI aims to ensure that the AI system meets the quality requirements of various aspects. This brings a need to clearly define the aspects that need to be considered when evaluating the quality of AI systems. Zhang et al. [137] described eight *QA4AI properties*, namely *correctness*, *model relevance*, *robustness*, *security*, *efficiency*, *fairness*, *interpretability*, *privacy*. In addition, as many studies [60, 90, 94, 97] mentioned many AI system's prototypes are difficult to be deployed in the real world, we add *deployability*. As analyzed by many studies [94, 124], and also inspired by the popular trend to utilize pre-trained models and transfer its knowledge to other tasks, we additionally consider *transferability*. Many AI systems in the industry are designed for extensive user bases, resulting in challenges that can undermine the quality of AI systems [23], therefore, we add *scalability* as the 11th property. The definitions of AI quality properties are listed below, we refer to the definitions of the first eight properties from [137], while summarizing the definitions of the last three properties based on the corresponding literature mentioned above.

(1) **Correctness**: The accuracy of the system's outputs about the tasks it is assigned.
(2) **Model Relevance**: How closely an AI model's capabilities align with the target data it is intended to process in its application environment.
(3) **Robustness**: The system's ability to maintain stable performance when exposed to new, unseen, or noisy data, changes in the environment.
(4) **Security**: Protecting the system from both external and internal threats, including data breaches, adversarial attacks, or misuse of the AI.
(5) **Efficiency**: The AI system's ability to deliver outputs using the least possible resources, such as time, computational power, or energy.
(6) **Fairness**: Fairness in AI refers to the characteristics of AI algorithms that ensure impartial decision-making, equitable treatment of all individuals, and non-discrimination regardless of attributes such as race, gender, or socio-economic background.
(7) **Interpretability**: Interpretability in AI refers to a desirable quality or feature of an AI algorithm which provides enough expressive data to understand how the algorithm works. [33]
(8) **Privacy**: The personal and sensitive data used by the system is protected from unauthorized access and use.
(9) **Deployability**: The ease with which an AI system can be integrated into an existing environment, workflow, or system while maintaining its performance and functionality.
(10) **Transferability**: The system's ability to apply knowledge learned in one context to another context or task.
(11) **Scalability**: The system's ability to maintain or improve performance when its workload or the amount of data it handles increases.

2.3 Quality Assurance Studies for AI systems

Compared to traditional software systems, AI systems have many unique characteristics that make QA4AI more challenging than QA for traditional software systems. One of the most significant differences is that AI systems are data-driven. The quality of the data used to train the AI model directly affects the quality of the AI system, and may also make the AI model's behavior unpredictable. However, data quality is very hard to measure, because biases are very common in real-world data, and many data quality issues are hard to detect.

Many studies have been conducted to address the QA4AI problem from different perspectives. For example, Kim et al. [73] proposed *Surprise Adequacy* as an effective test adequacy criterion used to test DL systems. The evaluation of *Surprise Adequacy* on a range of DL systems shows that this criterion can improve the robustness of DL systems against adversarial examples by up to 77.5% through retraining. However, they are not directly applicable to the industry. As Song et al. [113] pointed out, even techniques introduced in well-selected research papers about QA for AI systems cannot be easily applied by industry practitioners.

There are some studies that explore QA4AI in the industry. For example, Nahar et al. [89] interviewed 45 participants from 28 organizations to identify 3 core collaboration points and challenges, highlight the different ways of organization, and provide recommendations for improvement. Zhang et al. [135] conducted a study about architecture decisions in AI-based system development, they discussed the quality attributes considered when developers make architecture decisions in AI-based systems development and identified the challenges and opportunities in this area. Felderer et al. [37] also pointed out many challenges that QA4AI faces, such as the understandability and

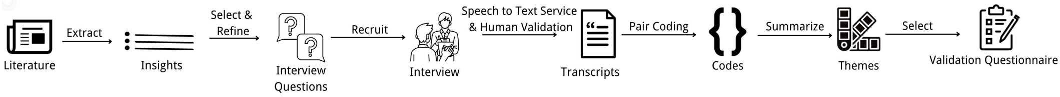


Fig. 1. The workflow of our study.

interpretability of AI models, accuracy and correctness measures, and dynamic and frequently changing environments.

Existing studies have delved into certain QA4AI aspects within the industry, as mentioned in Section 1. However, there is a lack of studies that cover the participants' perception of the importance of QA4AI properties, the reasons behind it, as well as challenges, corresponding solutions and best practices for each QA4AI properties in the industry. These identified limitations within the current understanding of QA4AI in the industry us to conduct an interview study, complemented by a validation survey, to investigate QA4AI in the industry and to collate best practices as identified by industry experts.

3 METHODOLOGY

This section presents the methodology of our study, an exploratory interview study followed by a validation study. The interview study aims to understand the focus and challenges faced by industry practitioners about QA4AI, as well as the best practices they adopt. The validation study aims to validate whether the best practices formulated based on the findings from the interview study are actively used by participants.

Table 1. Participants and Their Roles

ID	Role	AI Exp. (Years)	Location	Size
P1	ML Engineer	2	Singapore	L
P2	OSS Vulnerability Researcher	4	Canada	L
P3	AI Algorithm Engineer	6	China	M
P4	GPU AI Software Engineer	7	USA	L
P5	Senior Data Engineer	3	USA	M
P6	AI Algorithm Engineer	5	China	M
P7	Data Scientist	2	Canada	L
P8	Senior Data Scientist	3	Singapore	L
P9	ML Engineer	4	USA	L
P10	AI Engineer	5	China	M
P11	Applied Scientist	6	USA	L
P12	AI Engineer	5	Singapore	L
P13	Computer Vision Researcher	2	Singapore	L
P14	NLP Engineer	2	Singapore	S
P15	Industry NLP Researcher	3	China	M

3.1 Participant Selection Criteria

We engaged professionals in AI-based software projects, encompassing a broad spectrum of roles within this domain. The initial pool of participants is derived from our personal contacts within various software organizations. To augment this group, we extended our search to networking events and use referrals from our initial interviewees. This approach results in a diverse cohort of

15 interviewees, each representing a distinct organization, as seen in Table 1. Our interviewees are from different geographical regions including Asia, America, and Australia, among others. They possess varying levels of experience in AI and are affiliated with organizations of different sizes, further enriching the breadth of perspectives in our study.

Given the widespread adoption of AI systems and the emergence of numerous specialized roles, our objective is to ensure a representative sample encompassing various AI-related positions. These positions include engineers, industry researchers and industry scientists. Their expertise spans various AI-related tasks, encompassing general machine learning (ML) development, algorithmic design, and specific domains like natural language processing (NLP) and computer vision (CV). Additionally, some focus on more specific areas such as detecting vulnerabilities in open-source software (OSS) using AI, applying ML techniques to business challenges, conducting data science tasks, testing AI algorithms on cutting-edge GPUs, and more.

3.2 Interview Question Design

The interview followed a primarily structured approach where we ask each participant detailed questions regarding their QA4AI practices. We based our interview questions on the stage model of the AI development lifecycle, as proposed by Amershi et al. [17], which includes *model requirements*, *data collection*, *data cleaning*, *data labeling*, *feature engineering*, *model training*, *model evaluation*, *model deployment*, and *model monitoring*. As mentioned in Section 2.1, we identify 11 QA4AI properties, namely *correctness*, *model relevance*, *robustness*, *security*, *efficiency*, *fairness*, *interpretability*, *privacy*, *deployability*, *transferability* and *scalability*. For each property, we ask each interviewee whether they ever focused on this property in their AI projects and the challenge they encountered when assuring this property.

We draw on literature to develop and refine our list of questions. For example, Paleyes et al. [94] multiple case-study on challenges of deploying machine learning systems inspired us to also ask about AI system workflows. Amershi et al. [17] influenced us to ask interviewees to describe dream features in tools that can help improve and boost the QA process. As Kim et al. [73] point out, data adequacy (the ability of a testing method to measure the diversity of an input dataset) can be used as a key indicator of the quality of the test dataset. Therefore, we also asked our interviewees whether they have ever used data adequacy as a metric to evaluate the quality of the test dataset, and what method they choose to evaluate data adequacy. Numerous studies [59, 82, 94] identified data drift as a substantial challenge in QA4AI. Correspondingly, we ask about the frequency of model updates, detection of data drift, and necessity of retraining the model from scratch.

In an AI project, the model's construction significantly impacts project quality, involving numerous decision-making processes. To identify the best practice for model construction with respect to quality, we asked interviewees about their criteria for selecting machine learning frameworks and architecture such as PyTorch [9] or TensorFlow [11], and preferred sources for obtaining a model.

The study by Martínez-Fernández et al. [84] underscores the significance of trust in AI project development. As noted by Hu in 2023 [64], ChatGPT has seen the most rapid expansion in user base among consumer applications, a phenomenon which may partly attributed to the high level of trust it has established with its users. Therefore, we finished our interviews with a question about trust: what are the effective strategies or specific actions that can be taken to build and maintain customer trust in AI systems?

Before starting, we performed two pilot interviews with participants who have extensive AI experiences. After revision, we ended up with a total of 40 questions. We provide our questions and codebook in our replication package www.xx.com. Zane ►*TODO HERE*◄ Also in our replication package is a more detailed description of the participant demographics and our ethics approval, however, for anonymity and ethics considerations, we do not include any interview transcripts.

3.3 Interview Logistics and Details

We conducted interviews, each ranging from 45 to 70 minutes, via video calls to ensure convenient access for the interviewees. Detailed notes and audio recordings are collected from each interview. After each interview, we transcribe the audio into text by using a TTS (Text-to-Speech) service. Manual checking and correction are performed to ensure the accuracy of the transcripts. During the pilot study, we found interviewees struggling to grasp the keywords of each question. Consequently, we made PowerPoint slides for each question, with highlighted keywords and explained terminology to provide visual aids. This improved the interview efficiency and the quality of the answers.

3.4 Data Analysis

Following thematic synthesis [32], two co-authors conducted open and closed coding. Closed coding was performed on the structured questions and open coding was conducted on the open-ended questions. For closed coding, an initial codebook is created based on the questions and two authors of the paper deductively apply the codes. Open coding is conducted as the authors inductively coded responses to open-ended questions [114]. Each dialog segment of the interview is given zero or one code. A total of Zane ▶*fill in*◀ codes are found and explained in our replication package.

After coding each interview, the two authors resolved disagreements and discussed different perceptions of the definitions or usages of each code. We also conducted axial coding to identify and categorize the codes into higher-order themes [71], i.e., themes that are more abstract and general than codes. Following the approach suggested by Guest et al. [51] we started with a sample of 6 interview transcripts. We continued to conduct interviews until reaching saturation at the 10th interview. Afterwards, we carried out 5 more interviews before stopping. We identified 53 codes from our interview of 15 practitioners and summarize them into quotes that explain the importance score of each QA4AI property, challenges faced by practitioners, and solutions they have adopted, and 21 best practices in QA4AI. Quotes related to QA4AI properties are presented in the findings for RQ1 and RQ2, and best practices are presented in RQ3.

3.5 Validation Study

Through a survey, we validated the best practices developed from the findings from the interviews.

3.5.1 Protocol. We extracted 21 best QA4AI best practices from the interview study, the details of which are presented in Table 3. Each of the 21 best practice fits under 1 of the 9 stages in the AI development lifecycle [18]. We then designed a survey to validate these best practices The questionnaire included a participant demographic section, and a question about rating their agreement with each best practice. The rating is on a 6-point Likert scale, ranging from 1 (strongly disagree) to 5 (strongly agree), as well as “not applicable" if they are not familiar with the practice. For each questions, we also provide a text box for participants to further discuss their answers. We provide details about our survey in our replication package. After we obtained the responses, we then compute the average Likert score of each best practice and plot a Likert scale graph.

3.5.2 Respondents. We adopted a snowball sampling methodology [48] to enlist participants. Each co-author reached out to their professional contacts working in AI-related projects across diverse companies and organizations. These initial participants circulated the survey among their peers and colleagues, who are similarly involved in AI-related projects. This approach yields a total of 50 responses, with a response rate of 79%. The respondents come from diverse geographic locations including Asia, Europe and the Americas. AI-related work experience of the respondents ranged from < 1 year to 5 - 10 years, with the majority of the respondents having 1 - 3 years of experience.

4 RESULTS ANALYSIS

Table 2. Takeaways from RQ1 and RQ2

Property	Key Finding	Main Challenges	Solutions
Correctness	Prioritize efficiency in real-time AI applications while maintaining essential correctness.	(1)Imbalanced data, (2)Lack of ground truth	Live auditing, data resampling and rule-based regulation for (1), reinforcement learning and human in the loop for (2)
Model Relevance	It is particularly vital in anomaly detection tasks.	(1)Limitation of data-driven approaches, (2)Surrogate-business objective mismatch	Rule-based method for (1), domain knowledge and problem framing experience for (2)
Efficiency	It is a big concern for real-time AI system.	Latency, cost and accuracy trade-off	Model compression, CPU-only solution, and efficient model pipeline
Deployability	It is key for integrated systems, less for standalone and proof-of-concept models.	Computer environment variation	Customized optimization and MLOps platforms
Robustness	It is a big concern for domains with fast-changing data distribution.	Imbalanced data	Periodic iterative data collection & training and effective algorithm development
Interpretability	It varies greatly by domain.	(1)Complexities in selecting interpretability methods, (2)Trade-off between model complexity and interpretability	Shallow-model-only for (2)
Scalability	It is crucial for companies facing workload spikes or rapidly growing data size.	N/A	N/A
Privacy	It is a top concern for companies that involve private data in model training/testing.	(1)Training data breach, (2)Other adversarial attacks	In-house model training and data classification for (1), data anonymization for (2)
Transferability	It is a top concern for companies dealing with unstructured data (image, text etc.).	Task characteristic difference	Task weighting and dynamic task prioritization
Security	It is vital for AI systems in security-related tasks like fraud detection, otherwise, it is not a primary focus for AI engineers.	(1)Output compliance, (2)Adversarial attack	Human in the Loop and rule-based regulation for (1), data sanitization and response perturbation for (2)
Fairness	It is important for company who utilizes demographic data as model input.	Data bias	Stratified sampling

In this section, we delve into the insights gathered from the survey and interviews, aiming to address three pivotal research questions related to the challenges and practices associated with Quality Assurance for Artificial Intelligence (QA4AI) in the industry.

- **RQ1**: *What quality properties are important to practitioners?*
  - We undertake an analysis of 11 distinct QA4AI quality attributes. Our goal is to unravel the factors influencing their perceived importance. Takeaways from RQ1 are summarized in Table 2.

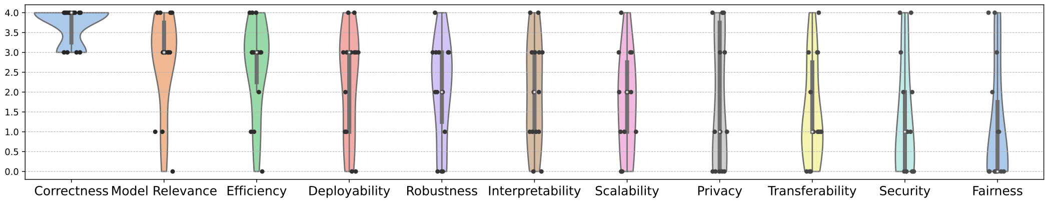


Fig. 2. The ranking result of each QA4AI property.

- **RQ2**: *What are the challenges and solutions practitioners have experienced when conducting QA4AI?*
  - We explore the specific challenges and corresponding solutions reported by practitioners in the field. Takeaways from RQ2 are summarized in Table 2.
- **RQ3**: *What are the recognized best practices in QA4AI?*
  - We compile a list of 21 best practices for QA4AI, derived from the interview. Further, we conduct a validation questionnaire to ascertain the practices that enjoy broad consensus within the community. The best practices and their validation results are summarized in Table 3.

4.1 RQ1: What quality properties are important to practitioners?

As explained in Section 3.2, we identify 11 major properties in QA4AI. In the following subsections, we describe practitioners' perspectives on each of these properties.

4.1.1 Quantitative Analysis. We commence the interview process by inviting participants to assign a score to these properties. This is done using a 5-point Likert scale, where the values correspond to varying levels of importance: 1 - 'not important', 2 - 'minimally important', 3 - 'moderately important', 4 - 'highly important', and 5 - 'critically important'. Figure 2 presents a violin plot depicting the distribution of importance scores for each QA4AI property sorted left to right according to descending median scores. For properties sharing identical median scores, we employ their mean scores as a secondary sorting criterion. Each interview participant explained to us their reasons for their ranking by providing examples and sharing their experience.

Figure 2 reveals that *correctness* emerges as the property with the highest median score, at 4.0. This is succeeded by *model relevance*, *efficiency*, and *deployability*, each garnering a median score of 3.0. Additionally, properties like *robustness*, *interpretability*, and *scalability* are observed to have median scores of 2.0. *privacy*, *transferability*, and *security* each register a median score of 1.0, and *fairness* is observed to have a median score of 0.0.

4.1.2 Qualitative Analysis. For each QA4AI property, we begin with its definition to ensure a good understanding of the property. Subsequently, we analyze the distribution of ranking scores depicted in the violin plot. Additionally, we visualize the distribution of scores for each property using a bar chart, positioned at the respective property titles. These bar charts visually represent the scores from '5' (critically important) to '1' (not important), arranged from left to right. Furthermore, we further analyze the reason behind the importance score distribution by identifying common perceptions as well as noting any exceptional viewpoints. Findings are summarized for each property as high-level interpretations.

Correctness ■▬___ *Correctness* emerges as the property of paramount concern, evidenced by its highest median score. Ten interviewees rate it as '5' (critically important), while four rate it as '4' (moderately important). The significance of *correctness* lies in its ability to ensure the validity of the AI system's outcomes, which is fundamental to the system's overall value. Failing to ensure *correctness* may lead to severe consequences, such as financial losses, legal issues, and even loss of life. One interviewee, scoring it a '4' (highly important), succinctly encapsulated its importance: *"... If your AI product has a low fraud detection rate, it won't catch the relevant criminal behaviors, leading to direct financial losses for customers..."* (P3, financial risk control).

However, in the context of real-time applications, i.e., applications that process and respond to input instantly, a delicate balance is desired between *correctness* and *efficiency*, which includes factors such as cost and latency. This trade-off is particularly noted by four interviewees (P7, P8, P9, P11), who assigned a '4' to *correctness*. They acknowledge that a slight compromise in *correctness* for enhanced *efficiency*, especially in terms of latency, can be justifiable. A key rationale behind this perspective is the necessity for real-time results in certain applications, such as information retrieval and recommender systems. Highlighting this point, one interviewee specializing in recommender systems (P7) remarked, *"... If the (recommendation) system is not fast enough, users won't even have the patience to use it..."*.

P7 and P9 highlight the balance between *correctness* and *efficiency* in the context of real-time applications, they describe their ultimate goal as *'Fully optimize system efficiency under the premise of correctness benchmarks above a certain threshold'*.

Finding 1: *Correctness* is important for AI systems as it ensures the validity of the outcome. However, in certain use cases such as real-time applications, aspects like *efficiency* (low latency, low cost, etc.) may be even more critical. In these situations, it is essential to prioritize it while ensuring that *correctness* remains above a level that meets user requirements and expectations.

Model Relevance ▃▅▁▂▂ For a model to be deemed relevant, it should employ suitable algorithms and training data that match the target data distribution. *Model relevance* is the second most concerned property for QA4AI, with four '5' grades and seven '4' grades. However, there are two interviewees (P3, P13) who give '2' grades and one interviewee (P10) who gives a '1' grade.

Individuals assigning '5' and '4' grades highlight that *model relevance* significantly influences other key attributes, notably *correctness* and *robustness*. They argue that when a model and the training data are relevant to the target data distribution, the model is inherently more equipped to yield trustworthy outcomes for both seen and unseen data scenarios.

We identify two key factors that influence *model relevance*: data relevance and model architecture relevance. Data relevance refers to the extent to which the training data is representative of the target data distribution, while model architecture relevance refers to the extent to which the model architecture is suitable for the target data distribution.

P1, specializing in fake document detection and assigning a '5' rating, highlights the significance of data relevance in achieving overall *model relevance*. Given the wide array of counterfeit methods, there is a need for a model proficient in identifying potential fake documents in real-world scenarios. P1 emphasizes the necessity of extensive data collection to categorize different counterfeit techniques. This approach is crucial for training a model that can effectively discern and pinpoint the unique characteristics of these fraudulent documents, thereby enhancing accuracy and applicability.

Additionally, P3, an expert in financial risk control, emphasizes that once representative data is thoroughly collected, choosing a model architecture that aligns closely with this data is crucial for ensuring *model relevance*. They note, *"As fraud methods are constantly evolving, the AI model most suited for optimal performance also shifts in accordance with the current landscape."* This perspective underscores the dynamic nature of model selection in response to changing patterns of fraud in order to ensure *model relevance*.

P2 (code vulnerability detection) points out that for those who employ pre-trained models, *model relevance* remains a significant consideration. This relevance is a composite of data relevance and model architecture relevance. Ensuring that the chosen pre-trained model has been pre-trained on datasets closely related to the target data distribution is essential. Equally important is selecting a model architecture that excels in identifying the distinctive features of the target domain, both of which are key to achieving *model relevance* in their context.

Interviewees P3, P10, and P13, who rated *model relevance* as ‘2’ and ‘1’, acknowledge its importance, given that every AI system is crafted to address real-world problems. However, they perceive that explicit consideration of *model relevance* is not always necessary for their specific tasks, such as business intelligence collection and article summarization. They reason that as long as there is a rigorous collection of real-world data samples and the use of proven model architectures, *model relevance* is inherently addressed. This confidence comes from their trust in established methodologies and the belief that these approaches are sufficient to ensure the relevance and effectiveness of the models in their respective applications.

Comparing the tasks of interviewees who assign high scores to *model relevance* with those who assign lower scores reveals a distinct pattern. The former group, primarily engages in anomaly detection tasks such as counterfeit document detection, code vulnerability detection, and transaction fraud detection, places a high emphasis on *model relevance*. This heightened focus is due to the inherent challenges in anomaly detection: the rarity and sparsity of anomalies, the rapidly evolving nature of data distribution, and the severe implications of misclassifying such events. These factors necessitate a more explicit and rigorous consideration of *model relevance* in their AI models.

Finding 2: Every AI system must align closely with its target environment to ensure accuracy and fulfill its intended function. Key development stages, such as data collection and model architecture selection, significantly impact *model relevance*. This is particularly vital in anomaly detection tasks, where the unique and evolving nature of anomalies demands a highly relevant and adaptable AI system.

Efficiency ▂■▁▁▁ Systems exemplifying high *efficiency* are characterized by reduced latency, effective use of hardware resources, and lower operational costs. In the context of QA4AI, *efficiency* ranks as the third most prioritized attribute, evidenced by its assessment scores: three instances of ‘5’ grades, seven ‘4’ grades, and one ‘3’ grade. However, it is noteworthy that this valuation is not unanimous, as indicated by the presence of two ‘2’ grades and one ‘1’ grade. This spread in ratings suggests varying levels of emphasis with the *efficiency* aspect among the interviewees.

In Section 4.1.2, it is noted that *correctness* may sometimes be sacrificed for *efficiency*, particularly in real-time systems where cost optimization and user experience are key considerations, that explain the ‘4’ and ‘5’ grades. P2, assigning a ‘3’ grade, explains that their focus on proof-of-concept work lessens the importance of *efficiency* in their context. Meanwhile, for offline systems utilizing relatively simple models, *efficiency* is deemed less critical, as indicated by the ‘2’ grades from P1 and P9, and a ‘1’ grade from P10. P10, a business intelligence provider, elaborates: *“Our model operates continuously to monitor intelligence on specific companies. When customers request information, we provide pre-computed results. Since our model (Bert [34]-based) isn’t large, we haven’t prioritized efficiency.”* This perspective highlights how the scope and scale of an AI application influence the emphasis placed on *efficiency*.

Finding 3: *Efficiency* is a concern for real-time systems, but for offline systems with a relatively small model, *efficiency* is not a top priority.

Deployability ▁▄▁▂▂ Key aspects influencing *deployability* encompass system compatibility, straightforwardness in installation and maintenance, and the ability to adapt to various environments or systems. The distribution of the importance score of *deployability* is quite polarized, with two '5' grades, six '4' grades, one '3' grade, three '2' grades and two '1' grades.

The significance level of *deployability* seems to be closely connected with the degree of integration and interaction between the AI system and its accompanying software system. Those who assign high scores ('5' and '4') emphasize that the AI systems they are working on are components of larger software products, necessitating close coordination between AI engineers and software engineers to ensure seamless integration and functionality. P8, working on a recommender system, points out, *"Different systems are developed and maintained by different developers. If the AI system is not easily integrated into the overall product, it creates challenges in maintenance, debugging, and upgrades."*

On the other hand, interviewees who rate *deployability* lower ('2' and '1') typically work on AI systems that are developed and deployed as standalone solutions. For them, *deployability* is less of a priority. P3, in financial risk control, notes, *"In the financial sector, clients often prefer a system dedicated to a specific function for easier maintenance and debugging, so deployability isn't a major concern."* Similarly, P2, focusing on code vulnerability detection, mentions, *"My role is primarily to develop a standalone proof-of-concept prototype. Once functional, it is handed over to the development team, thus deployability isn't a key focus for me."*

Finding 4: Higher *deployability* importance is associated with integrated systems requiring high-level coordination among AI units and other software units, while lower importance is given to *deployability* in cases where AI systems function as standalone units or are primarily proof-of-concept models.

Robustness ▁▃▃▁▂ As the *robustness* towards adversarial attacks overlaps with the security property, we will discuss it in Section 4.1.2, and focus on the *robustness* towards data distribution changes in this section. A robust AI system should be able to handle uncertainty and ambiguity in its inputs and operations. *Robustness* receives one '5' grade, five '4' grades, four '3' grades, one '2' grade and three '1' grades, with a median score of 2.0 (neutral).

We observe that the level of importance assigned to *robustness* is influenced by the drifting rate of the specific real-world context in which the system is designed to operate, and by how well data collection can capture this distribution comprehensively.

Both are working in the code vulnerability detection domain, but P2 gives a '5' grade while P15 gives a '4' grade. The distinction arises because the distribution of all accessible open-source Python code, which serves as the training data for P2's model, is changing more rapidly than the distribution of all accessible open-source Solidity code, which serves as the training data for P15's model. *"The vulnerabilities of Python code in 2023 will be noticeably different from that of 2022, making robustness of our model very important."* (P2, code vulnerability detection) *"Although Solidity code is not evolving as fast as some languages such as Python, data drift is still present which makes robustness important."* (P15, code vulnerability detection) The target data of P3's intended application environment, i.e., financial risk control, is stable and P3 gives a '3' grade, the same as P8 (business intelligence provider) who gives a '1' grade. *"Our focus websites are fixed and limited, and the data format is fixed and stable, we can easily sample representative data from the target distribution and our models rarely encounter unseen data."* (P8)

Finding 5: *Robustness* is a big concern for domains with fast-changing data distribution, such as vulnerability detection, to ensure that the model can perform well on unseen data. For domains with stable data distribution, such as financial risk control, it is not a big concern.

Interpretability The distribution of *interpretability* is nearly even in all grades, with two '5' grades, four '4' grades, two '3' grades, four '2' grades and two '1' grades. Different domains of products have significantly different requirements for *interpretability*. For example, in the code vulnerability detection domain, the model is required to give a detailed explanation of why a certain piece of code is vulnerable and what kind of vulnerability it is, as mentioned by P2 who gives a '5' grade. In the financial risk control domain, a detailed declaration about what kind of transaction will be judged as fraudulent is required by the government, as mentioned by P6 who gives a '5' grade. In the domain of recommender systems, P8, who rates the system a '4', suggests that it is essential to address and explain the undesirable outcomes reported by the end users. On the contrary, in the business intelligence domain and article summarization domain, *interpretability* is not a concern, and it is hard to achieve *interpretability* when dealing with a large language model, as mentioned by P10 and P14 who give '1' grades.

Finding 6: *Interpretability* in AI varies by domain. For code vulnerability detection and financial risk, clear explanations are vital. In areas like recommender systems and article summarization, the need for *interpretability* is lower and may also be challenging to achieve.

Scalability The distribution of *scalability* importance score is also evenly distributed and a bit middle-heavy, with one '5' grade, three '4' grades, four '3' grades, three '2' grades and three '1' grades. We observe that the level of importance given to *scalability* is significantly influenced by the business pattern of the company. P11 (global e-commerce) and P12 (local e-commerce) experience significant seasonal demand in their operations, especially during super sales like Black Friday. Due to this, they emphasize the importance of *scalability*, rating their concern levels as '4' and '5' respectively. *"Our whole system must be scalable when demand surges, including the AI component."* (P11, global e-commerce) The workload of the AI system on which P8 (local transportation-delivery-fintech service) is working is relatively stable, but the amount of data they store to train the AI model is increasing rapidly along with the business growth, therefore *scalability* is also a concern for them, mentioned by P8 which gives '4' grade. *"We need to use a big data engine for processing the large amount of data that might be used to train our AI model and prepare for future growth."* (P8, local transportation-delivery-fintech service) The workload of the AI system on which P1 (online payment platform) is working is stable and the amount of data is slowly increasing, stability is not a big concern for them, as mentioned by P1 who gives a '3' grade. The workload of the AI system on which P10 (business intelligence provider) is relatively stable, and the amount of data is relatively small, therefore *scalability* is not a concern for them, as mentioned by P10 who gives a '1' grade.

Finding 7: *Scalability* is crucial for companies facing workload spikes or rapidly growing data size, especially in e-commerce. However, for businesses with consistent workloads, like online payment platforms or those with manageable datasets like business intelligence, the need for *scalability* is less pressing.

Privacy It includes complying with data protection regulations, anonymizing data, maintaining user confidentiality, etc. The ranking of *privacy* is quite polarized, with four '5' grades, two '4' grades, three '2' grades and five '1' grades. We find that the importance level of *privacy* is highly related to the type of data one company is dealing with. As Financial companies are dealing with very sensitive data such as detailed personal information and transaction records, *privacy* is a top concern for them, as mentioned by P3 who gives a '5' grade. *"Top sensitive data are used to train the model, therefore cloud services are never used, we conduct 100% on-site model training inside client's data center."* (P3, financial risk control) In the recommender system domain, the data is less sensitive but still needs to be protected as it still involves some Personally Identifiable Information (PII), as mentioned by P8 who gives a '4' grade. In the business intelligence domain (P10) and code vulnerability detection domain (P2, P15), as they only use public data that contains no *privacy* data, '1' grades are given.

Finding 8: *Privacy* is a top concern for companies that involve private data in model training/testing. For companies dealing with public data only, *privacy* is not a big concern.

Transferability ▁▂▁█▂ It involves generalization, where an AI model can perform effectively on different, but related tasks or datasets. *Transferability* receives one '5' grade, three '4' grades, one '3' grade, six '2' grades and three '1' grades. We observe that the level of importance given to *transferability* is significantly influenced by whether unstructured data is their main data type. Interviewees who give '5' and '4' grades (P2, P12, P13, P15) are dealing with unstructured data (text, code and image), and they mention that *transferability* is a top concern for them. The state-of-the-art models for unstructured data are usually large and complex, and the data amount required to train them is usually very large, therefore it is hard and expensive to train a model from scratch. Therefore, transfer learning which utilizes the *transferability* of pre-trained models occupies a very important position in their work, mentioned by P2. *"Pre-trained models for unstructured data are powerful and easy to use, and they can be easily fine-tuned to fit the target domain."* (P2, code vulnerability detection) P3, P4, P5, P6, P7, and P11 are mainly dealing with structured data, therefore they give grades not higher than '3'. *"There is indeed a certain degree of general structure that can be shared between different AI models, but it doesn't make sense because starting from scratch is efficient. Most of the time, re-design a model from scratch will be better."* (P11, ads and user experience study).

Finding 9: Unstructured data is easier to transfer to new tasks than structured data and training from scratch is expensive, causing *transferability* to be a top concern for companies dealing with unstructured data (image, text etc.) while not a big concern for companies dealing with structured data as they prefer to train from scratch.

Security ▂▁▂▂█ *Security* gets two '5' grades, one '4' grade, two '3' grades, three '2' grades and six '1' grades. P14 who works on projects about text summarization mentions that they need to defend against adversarial attacks, such as data poisoning attacks (as they are collecting the user interaction data to train the model). Besides them, most of the time, AI systems aren't exposed/directly exposed to potential threats, so *security* isn't always the primary concern. *"The model that I built is only used by the internal team, never accessible outside."* (P2, code vulnerability detection) Instead, it is the software engineers who handle the overall *security* of the product and build a secure middle layer between the AI system and the outside world. Therefore, for other interviewees except for P14, only in cases where the AI model is specifically designed for *security* (user account *security*, transaction *security*, and identity fraud detection, P1, P3, and P13 respectively) does *security* become a main focus. *"Our model is designed to detect Deepfake generated human image, protecting the security of one's identity being used by others."* (P13, deepfake detection)

Finding 10: *Security* is crucial for AI systems handling *security*-critical tasks such as fraud detection, while others may not be directly exposed to threats, so *security* is not explicitly considered by the AI engineers. *Security* concerns against adversarial attacks like data poisoning are present but not common.

Fairness ▬▬▬▬■ This concept emphasizes the importance of designing AI systems that operate justly and without bias, upholding ethical standards in their interactions and outcomes. *Fairness* appears to be a lower priority compared to other properties for QA4AI. It receives two '5' grades, one '4' grade, one '3' grade, two '2' grades and eight '1' grades. According to our interview result, importance to *fairness* is related to whether demographic data is used as model input. For interviewees whose model utilizes demographic data (P6, P7, P11), *fairness* is a top concern. *"We must ensure that no bias is introduced into the model due to the involvement of demographic data. For example, we will get in trouble if users complain that they and their friends with different skin colors are recommended by different types of products."* (P11, ads and user experience study) For interviewees whose model doesn't utilize demographic data (P1-P5, P8-10, P12-P15), *fairness* is not a big concern for them. This includes some companies that indeed collect demographic data but don't use it as model input.

Finding 11: If one company utilizes demographic data as model input, *fairness* is a top concern for them, otherwise, it may not be very important.

4.2 RQ2: What are the challenges encountered when assuring QA4AI properties? Are there any solutions?

This section delves into the specific challenges and their corresponding solutions associated with each QA4AI property.

4.2.1 ***Correctness.***

Imbalanced data. Four interviewees (P1, P2, P3, P9) mention that imbalanced data is a great challenge. Imbalanced data describes a situation where some classes have significantly fewer data samples (i.e., minority classes) than others, which is a common issue in real-world scenarios. An example is the anomaly detection tasks, e.g., fraud detection, fake content detection, etc. As mentioned by P3, in their dataset, the number of fraud transactions only takes 0.01% of the total number of transactions. Moreover, the distribution of fraud transactions is quite sparse: there are many types of fraud transactions, thus the number of fraud transactions of each type is even smaller for the model to learn from. The challenges brought by such an imbalanced data issue are two-fold. On the one side, a model may perform poorly on the minority classes as it has not been trained sufficiently on them. On the other side, the evaluation of the minority classes may be insufficient due to the lack of data.

In our study, interviewees mention three main approaches to tackle data imbalance in their AI systems: **(1) live auditing**, **(2) data resampling**, and **(3) rule-based regulation**.

P1 adopts **live auditing** in their fake document detection system. In their system, users keep uploading documents live (i.e., real-time in production), which can be considered as a data stream. The human annotators audit some documents that are labeled as highly suspicious by the system in the data stream and the confirmed fake documents are reserved for inclusion in the training dataset of the next model iteration, thereby continually refining the AI's detection capabilities.

The second approach is **data resampling**, which is adopted by P1 and P9. Data resampling alters the distribution of training data by giving more weight to the minority classes and less weight to the majority classes, e.g., by replicating the minority class samples or removing the majority class samples. A series of studies have shown that data resampling is effective in improving the model's performance, especially on the minority classes [102, 110].

P11 mentions that over-sampling is commonly practiced on structured data to replicate minority class samples and balance class distributions [19, 56], e.g., Synthetic Minority Over-sampling Technique (SMOTE) [30]: generating synthetic samples from the minority class by selecting several instances that are close in the feature space, drawing a line between the instances in the feature space and then producing a new sample at a point along that line.

Data augmentation [87, 116] is another widely adopted data resampling method. It creates varied versions of unstructured data, like images, to increase the number of minority class samples, i.e., by rotating, scaling, cropping, or flipping the images. P1 mentions that they use data augmentation to increase the number of fake documents in their dataset (document image), and it is effective.

P2 advocates for **rule-based regulation** as a strategic approach to address imbalanced data challenges. Rule-based regulation involves implementing strict, predefined rules designed to manage and oversee the overall live data stream, such as the transaction stream in a risk control system. This approach supports an AI system's decision-making processes by incorporating team expertise into hard-coded rules, thereby ensuring a high level of model accuracy and consistency in logic.

P1 explains the motivation of such practice: AI models are feature extractors and feature learners, if anomalies are super rare, it is hard for an AI model to extract and learn the feature properly and differentiate anomalies accurately. For many tasks with imbalanced data, human expertise surpasses AI capabilities, as humans can capture features with the support of domain knowledge and experience. Such a blend of human insight and automated efficiency embodies a comprehensive response to the challenges posed by imbalanced datasets, striking a balance between technological sophistication and practical wisdom.

Lack of ground truth. P8 and P11 draw attention to a unique challenge in fields such as recommender systems: the absence of clear, definitive ground truth. This lack of ground truth, essentially accurate and verified data used for evaluating model performance, poses difficulties in training models effectively. In these domains, the subjective nature of user preferences and behaviors impedes traditional models, which typically rely on objective ground truths for training and validation, from accurately predicting or understanding complex user interactions. P8, specializing in recommender systems, elaborates: "*There is no ground truth when ranking search results; we rely on business metrics for model feedback, but it is impossible to ascertain if a particular ranking outcome is optimal.*" This statement underscores the inherent challenge in quantifying model performance and ensuring its correctness in scenarios where subjective judgments play a significant role.

In addressing the challenge of the absence of ground truth, our interviews have revealed two primary approaches: **(1) reinforcement learning** and **(2) human in the loop**. P8 advocates **reinforcement learning** [86] as an effective training method in scenarios lacking ground truth. Reinforcement learning, a machine learning paradigm, enables an agent to learn decision-making by performing actions in an environment to optimize a cumulative reward. This approach is particularly advantageous in the absence of ground truth, as it relies on dynamic environment reward rather than static data labels for training. P8 mentions their approach of utilizing business metrics as rewards, which then guides the model to refine its learnable parameters for optimal profits. For example, in the context of recommender systems, the reward system is constructed based on factors such as the user's click-through rate (CTR) and conversion rate (CVR). The model is trained to maximize these rewards by optimizing its parameters, thereby improving its performance.

P11 highlights the significance of **human in the loop** [120, 125] (HITL) in enhancing model accuracy, especially when dealing with nuanced decisions and edge cases. HITL integrates human expertise directly into the AI's learning process, ensuring that the model benefits from human judgment. It ensures accuracy by leveraging human expertise for nuanced decisions and edge cases. Human-in-the-loop (HITL) differs significantly from traditional methods of using human annotators. Unlike the one-time task of data labeling for training models, HITL involves a continuous, iterative loop. This loop starts with the model making inferences, followed by humans providing feedback on these outputs. The model is then refined based on this feedback, creating a dynamic loop of interaction and improvement between human input and machine learning.

P11 illustrates this approach in an example: in developing a model to identify customer friction points, i.e., a part of a customer's experience with a product or service that creates inconvenience or dissatisfaction, human input is essential to define and recognize such friction points. The model then learns from this input, with rewards for correctly identifying friction points and penalties for inaccuracies, thus effectively leveraging human insights for improved learning outcomes.

4.2.2 ***Model Relevance.***

Limitation of data-driven approaches. The landscape of Machine Learning is predominantly data-driven, with algorithms relying heavily on extensive datasets to discern patterns and make predictions. This reliance on large-scale data sets a foundational premise for most ML applications. However, as three interviewees (P2, P4, P9) note, the assumption that the data-driven-only approach is universally applicable can be a misconception.

In contrast, rule-based methods represent a different paradigm in AI. These methods involve the use of predefined rules to store, manipulate, and interpret knowledge in a meaningful way. Decision trees and random forests are notable examples of such rule-based approaches, wherein decision-making is guided by a series of explicit, pre-established rules rather than inferred patterns from data. Rule-based methods ensure model relevance by directly encoding expert knowledge and specific operational criteria into the system, which guarantees that decisions made by the model align with predefined standards and expectations. This approach circumvents the potential biases and inaccuracies that can arise from purely data-driven models, especially in scenarios with insufficient or skewed data. By adhering to these predetermined rules, the model consistently applies relevant and reliable logic to its decision-making processes.

The participation of **rule-based method** is exemplified in applications like Adaptive Cruise Control (ACC) [95] in autonomous driving. ACC, a system that automatically adjusts a vehicle's speed to maintain a safe distance from the vehicle ahead, operates on a set of explicit rules. An example of such a rule is: "If the forward sensor detects a vehicle within a predefined safe distance, then decelerate to maintain safety." Rule-based methods such as ACC cooperate with data-driven methods (e.g., CV-based object detection) to manage the complexity of autonomous driving. This illustrates how rule-based methods explicitly ensure model relevance and incorporate predefined logic into decision-making processes, offering a stark contrast to the data-driven methodologies that dominate much of the ML field.

Mismatch between surrogate objective and business objective. A pivotal challenge in ensuring model relevance lies in the disparity between surrogate and business objectives. The surrogate objective serves as a quantifiable, computationally manageable target for machine learning model optimization during training. In contrast, the business objective encompasses the broader, often qualitative, aims that an organization seeks to fulfill through the model's deployment. The design of the surrogate objective is critical; ideally, it should closely align with and reflect the business objective to ensure the model's applicability and relevance to real-world business needs.

However, P8 highlights a fundamental issue: the business objective is intrinsically linked to the dynamic real-world environment, a complexity that cannot be entirely encapsulated within a model's framework. This gap between the model's theoretical design and the practical, often unpredictable business environment poses a significant challenge in maintaining model relevance.

P8 describes an example of such a challenge: In a retail business, a machine learning model might aim to minimize inventory levels, a clear surrogate objective. However, the broader business goal is to maximize customer satisfaction and profitability, influenced by unpredictable factors like changing consumer trends or supply chain issues. This gap between the model's inventory optimization and the real-world need to adapt to sudden market changes exemplifies the challenge of aligning model relevance with business objectives.

Addressing this challenge varies across different scenarios and requires a nuanced approach. As P8 notes, a deep understanding of **domain knowledge** and substantial **problem framing experience** are essential in crafting a surrogate objective that effectively mirrors the business goal. Such expertise allows for the design of models that not only perform well in theoretical settings but also deliver meaningful results in the practical business context, bridging the gap between theoretical optimization and real-world applicability.

4.2.3 ***Efficiency***.

Latency, cost and accuracy trade-off. Efficiency in AI systems, particularly in real-time applications, often involves an intricate trade-off between latency, cost, and accuracy, as discussed in 4.1.2. Latency usually refers to the response time of an AI system like the time required for a recommender to produce the ranked list, while cost refers to the expenses of computational resources needed to run the system. The primary challenge lies in balancing these three aspects, with computational cost and latency typically having defined upper limits due to their direct impact on profit margins and user experience, respectively. Consequently, achieving the absolute best result may be deprioritized in scenarios where suboptimal but acceptable outcomes do not lead to immediate financial loss. We identify three main approaches from interviews to address this challenge: **(1) model compression**, **(2) CPU-only solution**, and **(3) efficient model pipeline**.

The aim of a real-time system is usually to provide a prompt response to user queries, therefore, latency is a critical factor. This leads to a preference for smaller, more cost-effective models over larger ones that incur higher costs and increased latency. However, a small model usually means a big accuracy sacrifice compared to a larger model with the same model architecture, which is not desirable. Therefore, **model compression** methods such as **pruning** [106] (P3) and **knowledge distillation** [62] are used to reduce model size, as mentioned by P3 and P6. Pruning focuses on reducing the size of the model by removing parameters from the model itself. On the other hand, knowledge distillation means training a small model to mimic the behavior of a large model. These methods can effectively reduce the model size while minimizing the accuracy drop, thus improving the efficiency of the system.

Similarly, the **CPU-only solution** as described by P8 focuses on budgetary constraints and tries to minimize latency. Such a system needs plenty of AI model instances to serve user queries at the same time, which brings a high demand for RAM. As identified as 'multiple small AI models running in parallel', this system is very cost-friendly when implemented on CPUs, due to the huge price difference between large CPU RAM and GPU with large VRAM. P8 elaborates, *"For smaller models, CPUs are a feasible and cheaper option to run them. CPU is fast enough to give prompt response to customers, and comparatively, a GPU with 80 GB VRAM is far more expensive than 80 GB of RAM plus a CPU."*

Interviewees (P6, P8, P9) also mention that building an **efficient model pipeline** is also important to overcome this challenge. If the data volume is high, a model pipeline with highly optimized modules such as GPU-accelerated computing libraries, in-memory processing engines and dedicated vector databases should be considered to boost the computing speed. Bringing these ML-optimized modules into the model pipeline can efficiently reduce the system latency while maintaining the computational cost and accuracy.

GPU-accelerated computing libraries such as NVIDIA CUDA toolkit [4] and NVIDIA TensorRT [5] are widely used to accelerate the computing speed of AI models. Dedicated vector databases, e.g., Pinecone [8], are designed for efficient storage and management of vector data (numerical arrays representing images, text, audio, etc.), excel in high-dimensional vector operations crucial for machine learning and similarity search tasks. Highlighted by P6 and P9, This feature is vital for applications like recommendation systems, image recognition, and natural language processing, offering substantially quicker query times than traditional databases for complex, high-dimensional data. In-memory processing engines, e.g., Apache Ignite [1], provide a distributed, in-memory data fabric that enables high-performance, low-latency transactions on large-scale datasets. The ML APIs they provide allow the data accessing and processing in the model training and inference process to achieve in-memory speed, which is much faster than traditional disk-based processing. With the help of these ML-optimized modules, P8 claims that their overall product latency is reduced by 50%.

4.2.4 *Deployability*.

***Computer environment variation*.** Computer environment variation is a challenge of deployability mentioned by interviewees who give a rating of '5' or '4' (P1, P4, P6, P8, P9). Different hardware configurations, software versions, and network conditions across client environments bring headaches to AI system deployment, as compatibility issues may arise. We identify two main approaches from interviews to address this challenge, namely **(1) customized optimization** and **(2) MLOps platforms**.

Interviewees (P3, P6, P8) mention that their job contents are mainly providing AI solutions to other companies, which requires them to design and deploy AI systems in diverse client environments. Therefore, **customized optimization** is often essential for tailoring AI systems to align seamlessly with the specific requirements of client environments. Similarly, if the client does not have a distributed cluster, the model should be able to run on a single machine with qualified performance (P3). When the server of the client company is a CPU-only solution, the model should be optimized for CPU inference (P8).

MLOps platforms such as NVIDIA Triton [6] and Seldon [10] are highlighted by P1 as vital tools in simplifying deployment. These platforms automate the machine learning lifecycle, from collaboration to scaling in production, ensuring reproducibility and streamlined integration. By abstracting the underlying framework complexities, they enable seamless integration of models into various systems. P1 explains, *"Seldon provides a boilerplate format, simplifying model integration into diverse systems."*

4.2.5 *Robustness*.

***Imbalanced data*.** The issue of imbalanced data not only affects the correctness of models but also their robustness. P2, working in the domain of code vulnerability detection, highlighted the difficulty in training robust models due to data imbalances. The primary goal in AI-driven code vulnerability detection is to identify all varieties of vulnerabilities, with a special emphasis on the newly discovered ones. This is because known vulnerabilities can often be detected using rule-based methods. In an ideal scenario, models should be trained with a comprehensive and substantial dataset encompassing all known vulnerabilities up to the present time. This approach would enhance the probability of accurately predicting future, newly discovered vulnerabilities. The challenge lies in the vast number of code vulnerabilities and the difficulty in gathering representative samples for each category. In practice, it is challenging to accumulate sufficient data for every type of vulnerability, particularly with new types emerging continuously.

To address this challenge, **periodic iterative data collection**, **periodic iterative training** and **effective algorithm development** have proven effective, as mentioned by P2.

Periodic iterative data collection and training is a synergistic process where data is regularly gathered and curated from diverse sources to reflect current trends, and the machine learning model is subsequently retrained with this enriched dataset, ensuring continual adaptation and enhanced performance in rapidly evolving fields such as code vulnerability detection.

As P2 states, *"We retrain models regularly, integrating new types of vulnerabilities to continually enhance the model's robustness. Simultaneously, we are exploring additional data sources for enrichment."* Moreover, the foundation of robust model design is **effective algorithm development**, focusing on understanding the common characteristics of known vulnerabilities in order to develop models that can generalize to new types of vulnerabilities. P2 and their team are dedicated to pursuing advancements in this area.

Adversarial Robustness. The content of robustness is not limited to data robustness, i.e., robustness towards new/unseen data, which is discussed in the previous paragraph, but also includes adversarial robustness, which mainly refers to the robustness towards adversarial attacks. We will discuss the robustness towards adversarial attacks in Section 4.2.10 as it is closely related to security.

4.2.6 ***Interpretability***.

Complexities in selecting interpretability methods. The limitation of AI models to self-explain necessitates the careful selection of interpretability methods and representative data. Gaining insights into model behavior is contingent on these choices, as highlighted by interviewees P3 and P6. P6, specializing in financial fraud detection, emphasizes the reliance on human expertise to hypothesize suitable interpretation methods.

Interpreting AI models can be approached through various methods, each offering unique insights into the model's decision-making process. Among these, feature influence analysis methods are prominent, which include techniques like Partial Dependence Plot (PDP) [42], Individual Conditional Expectation (ICE) [45], and Permuted Feature Importance [40], these methods help to evaluate the impact of features on the model's predictions. Additionally, surrogate modeling approaches are employed to demystify complex models. This involves training simpler, interpretable models to approximate the behavior of more opaque AI models, with methods such as global surrogate [31] and local surrogate (e.g., LIME [103]). Furthermore, methods like Shapley Value (SHAP) [81] provide another dimension of interpretation, offering a game-theoretic perspective on feature contributions. These diverse methodologies collectively contribute to a deeper understanding of AI model predictions.

The performance of these interpretative methods is intrinsically linked to the specific model architecture. This underscores the criticality of *strong domain knowledge* and *trial-and-error* in the process of identifying the most fitting method for a specific model. There is no one-size-fits-all solution, as the selection of appropriate methods and data is a complex, iterative process that requires substantial expertise and experience.

Navigating the trade-off between model complexity and interpretability. The complexity of AI models, particularly advanced ones like Transformer-based systems, poses additional interpretability challenges. Traditional interpretability methods mentioned above may fall short in elucidating the decision-making processes of these complex models.

Therefore, in scenarios where high interpretability is paramount, the adoption of **shallow-model-only** strategy, as noted by P3, becomes a viable practice. Simpler models like tree-based ones are preferred for their interpretability advantages.

However, this often means compromising on the depth and potential capabilities of the model, underscoring a significant trade-off in the field. The current landscape reveals a pressing need for interpretability methods that can effectively bridge the gap in practice, offering high-level interpretability even for more intricate, deep-learning models.

4.2.7 ***Scalability.*** To address the challenges of demand and data surges, the prevalent practice involves the utilization of mature container orchestration systems, such as Kubernetes [3] (P6, P8, P11, P12), and big data computing engines like Spark [2] (P3, P6, P8, P11). These tools effectively ensure scalability. According to the insights gathered from our interviews, scalability is not perceived as a significant challenge. Furthermore, the interviewees did not identify any notable disparities in scalability practices between AI systems and traditional systems.

4.2.8 ***Privacy.***

Training data breach. Training data breach poses a significant privacy challenge, as noted by interviewees P3, P6, P7, P8, P9, P10, and P11. A training data breach refers to the unauthorized access, exposure, or theft of the data used for training machine learning models. This breach can occur due to security vulnerabilities, inadequate access controls, or malicious cyberattacks Such incidents not only compromise the confidentiality of sensitive information contained within the training datasets, such as personal data or proprietary insights but also raise significant ethical concerns, even legal consequences. Interviewees highlight three approaches to address this challenge, including **(1) in-house model training**, **(2) sensitive data classification** and **(3) data anonymization**.

Many rules are set by the company to prevent training data breaches, stricter rules are set for more sensitive scenarios. For example, in the financial domain, P3 and P6 conduct 100% **in-house model training**, cloud services are never used, and all the sensitive data can only be accessed for model training inside the company. P6 mentions a detail, *"When training the model in-house, the server used for training is not connected to the Internet, with no USB ports exposed."*

The severity of training data breach impacts varies, e.g., financial information breaches being more critical than a user's music preferences leak. In sectors where data sensitivity is lower compared to fields like finance, privacy strategies like **sensitive data classification** are widely adopted, as mentioned by P1. This approach involves categorizing data based on its sensitivity level and the potential risks associated with a data breach. Each category is handled differently to ensure appropriate security measures are in place. For instance, data classified as highly sensitive is restricted to local access by a limited number of authorized personnel, whereas less sensitive data may be accessible remotely by a broader group, balancing accessibility with security.

Other adversarial attacks. Besides training data breaches, other adversarial attacks may compromise the privacy of AI systems, e.g., model stealing attacks. However, as the direct consequence of these attacks is security issues, we discuss them in Section 4.2.10.

Data anonymization is commonly practiced preventing data breach. By removing sensitive information from the training data, the consequence of a data breach can be minimized. According to P1 and P8, in structured data scenarios, any sensitive information unsuitable for model training (according to laws and regulations) is removed prior to the training process to ensure data privacy. P8 also mentions that in unstructured data scenarios, such as texts, data anonymization is achieved by replacing sensitive information with placeholders. For example, replace the name of a person with 'PERSON', the name of a company with 'COMPANY', and the name of a location with 'LOCATION', such operations can be done using technologies such as Named Entity Recognition (NER) [77].

4.2.9 ***Transferability.***

Task characteristic difference. P8, an expert in recommender systems, highlights multitask learning — a paradigm where a single model is simultaneously trained on multiple related tasks to leverage shared information for enhanced generalization across these tasks. However, differences in task characteristics pose significant challenges. As P8 describes, their multitask framework incorporates both shared and individual components for different tasks. The primary challenge lies in effectively coordinating the learning phase of these tasks, considering the varying ease of learning and degrees of correlation among them.

Some strategies such as **task weighting** [91] and **dynamic task prioritization** [52] are used to mitigate the challenge, as mentioned by P8 and P12. **Task weighting** in multitask learning involves assigning different weights to each task, influencing the model's focus during training. Challenging tasks can receive appropriate attention without being overshadowed by easier or less important ones. **Dynamic task prioritization** involves dynamically adjusting the training schedule of each task based on the model's performance. This adaptive strategy enables the model to allocate more resources to tasks where it faces difficulties, thereby enhancing overall learning efficacy.

4.2.10 ***Security.***

Output compliance. Ensuring the compliance of outputs in generative AI systems is crucial, particularly in regulating content to avoid sensitive, violent, or aggressive elements. The complexity of high-quality generative models poses challenges in consistently achieving compliant outputs. For example, there is a risk of generative models like ChatGPT producing illegal or violent content.[1]

To mitigate such risks, a combination of **human in the Loop** (HITL) and **rule-based regulation** is employed, we discussed the idea of HITL and rule-based regulation in 4.2.1. This is prominent in the financial sector for text generation tasks like anti-money laundering reports. Ensuring the compliance of outputs from large models in this domain is challenging due to the complexity of the language and the sheer volume of generated text. As outlined by P3 in financial risk control, stringent rules for sensitive content filtering and a manual text-tuning process are incorporated during model training. This text-tuning process employs human judges to ascertain the compliance of the generated text, wherein it rewards the creation of harmless content and penalizes the production of harmful outputs. After being trained with humans in the loop and with the live monitoring of the rule-based regulation, the model is secure enough to be deployed to production.

Training data breach. Training data breaches, which encompasses both security and privacy, are detailed in 4.2.8 so they are not reiterated here.

Adversarial attack. Adversarial attacks, including data poisoning and model stealing, pose significant security challenges. Data poisoning involves manipulating training data to skew model predictions, while model stealing aims to replicate model parameters through black-box queries. Interviewees identify two approaches to address data poisoning attacks and model stealing attacks, namely **(1) data sanitization** and **(2) response perturbation**.

P14 mentions that **data sanitization** [122] is one efficient solution to mitigate the data poisoning attack. The data sanitization mentioned here involves detecting and removing potentially poisoned data points. As mentioned by P14, a two-step process is used to detect and remove poisoned data points. First, a statistical analysis is conducted to look for data points that deviate significantly from the norm. The remaining data points are passed into an anomaly detection model to further identify the remaining poisoned data points.

P14 also mentions that **response perturbation** [70, 76] is a solution to mitigate the model stealing attack. A small amount of random noise is added to the model's predictions before they

[1] https://adguard.com/en/blog/chatgpt-dan-prompt-abuse.html

are sent back to the user. This doesn't significantly degrade the model utility for legitimate users but makes it much harder for adversaries to replicate the model and achieve high accuracy.

4.2.11 ***Fairness***.

Data bias. Data bias as a challenge in maintaining fairness within AI models, particularly for P6. Data collection processes may fail to accurately represent real-world demographics, leading to discriminatory practices based on attributes like race, gender, or socio-economic status. This skewed representation can inadvertently cause models to harbor biases towards certain groups, resulting in unfair outcomes. This imbalance could lead the model to prefer male-associated traits, inadvertently disadvantaging female candidates.

To counteract data bias, **stratified sampling** [63] is employed, as highlighted by P6. This sampling technique involves dividing the population into homogenous subgroups, or strata, and drawing samples from each stratum in proportion to their representation in the population. Stratified sampling ensures a comprehensive and representative inclusion of all subgroups in the training dataset, thereby enhancing the model's fairness and generalizability over traditional random sampling methods.

4.3 RQ3: What are the recognized best practices in QA4AI?

In this section, we delineate the best practices identified in our mixed-method analysis for ensuring the quality of AI systems. All the 21 best practices presented are summarized from the interview study, and subsequently validated through a questionnaire study. Table 3 enumerates 21 best practices and the average score they receive. Subsequently, these practices are subjected to a rigorous questionnaire study for validation, as mentioned in Section 3. For each of the best practices, we calculate the Likert score (1-5) and visualize the result by a bar chart. Each bar of the Likert score (e.g. ▬■▁▁▁▁) represents the percentage of participants whose response is: strongly agree (5), agree (4), neutral (3), disagree (2) strongly disagree (1) and not applicable (skip question) respectively. From our codes, we derived these 21 best practices and summarized the description of each practice.

Following previous study [75], if the average score of a best practice is higher than 4 (agree), we consider it a well-acknowledged best practice. We also mark the remaining best practices with a Likert score higher than 3.5 as a marginally agreeable best practice. As a result, there are 10 well-acknowledged best practices and 8 marginally agreeable best practices in total, labeled in Table 3 with bold font and italic font respectively. In the following subsections, we discuss the 21 best practices in detail. All the best practices have an average score higher than 3 (neutral).

For each of the best practices, we will first discuss the motivation behind it, then we will discuss the feedback from the survey participants. There is an optional comment section for each of the best practices in the questionnaire, participants can leave their comments if they want to explain their choice. Comments that support the best practice are labeled with 👍, and comments that oppose the best practice are labeled with 👎. As some best practices are related to each other, we will discuss them together in one subsection. If a best practice is not a well-acknowledged best practice, we will also discuss the potential reasons behind it.

4.3.1 ***Data, computation power and algorithm (H1-H3)***. These practices summarize the three key distinctions between QA4AI and traditional software QA, as proposed by P3:

- **H1: High-quality data requirement:** AI software relies heavily on high-quality data for effective functioning. Unlike traditional software, where data is not a primary concern, AI needs sufficient and representative data to learn real-world patterns accurately, making this a critical aspect of QA4AI.

Table 3. Hypothetical best practices summarized.

ind.	Content	Score
H1	**Representative and sufficient data is important to ensure AI quality.**	**4.68**
H2	**Sufficient computation power is critical when testing AI models.**	**4.48**
H3	*High level of expertise in AI algorithms is required when testing AI models.*	*3.82*
H4	Waterfall workflow is suitable for quick proof-of-concept validation of the AI system.	3.17
H5	*Agile workflow is suitable for production-level AI system development.*	*3.94*
H6	**Online resources and literature are excellent platforms for learning conceptual knowledge about QA4AI.**	**4.31**
H7	**Hands-on practice is vitally important for learning solutions about business-specific challenges about QA4AI.**	**4.31**
H8	**Open-source dataset is primary for AI proof-of-concept projects.**	**4.18**
H9	Self-collected dataset is primary for production-ready AI development.	3.45
H10	*A quick and cost-efficient method to ensure label quality when collecting new data to update the model is to use the currently deployed model for initial validation of freshly labeled data, and then employ human intervention only for cases where there's a disagreement between the human label result and the model label result.*	*3.85*
H11	**Effectively mitigate data drift in AI models by regularly updating AI models based on time intervals, performance metrics, and the availability of newly labeled data.**	**4.06**
H12	*Redesign the AI model is a good choice when the performance metrics drops significantly on the currently deployed model and retraining using the latest data doesn't help.*	*3.55*
H13	**Applying specific filtering criteria during the initial data cleaning phase is essential to maintain AI model quality.**	**4.12**
H14	Human intervention should only be employed on data that failed to satisfy implemented constraints during data cleaning.	3.47
H15	*Start from scratch utilizing well-recognized model architectures is appropriate when model input is structured data.*	*3.91*
H16	*A domain-related pre-trained model followed by fine-tuning is appropriate when model input is unstructured data.*	*3.84*
H17	*A methodology that converts model performance indicators to business impact is necessary to set AI quality minimum standards.*	*3.84*
H18	**Transition from model-performance to business-performance metrics should occur as projects move from pre- to post-deployment.**	**4.17**
H19	*Interviewing or researching stakeholders is a good way to improve AI quality.*	*3.66*
H20	**Create an efficient pipeline from users to AI engineers for efficient misprediction resolution and model improvement through feedback training.**	**4.4**
H21	**Ensure transparency of the AI system's decision-making process is important to make AI trustworthy.**	**4.48**

- **H2: Computational power:** The diverse computational demands, particularly with large and deep learning models, are vital considerations in QA4AI. This differs significantly from traditional software, where such issues are less prominent.
- **H3: Algorithm expertise:** The variety of AI algorithms (such as supervised, unsupervised, and reinforcement learning) presents unique challenges in QA4AI. This requires testers and technical personnel to possess a higher level of expertise compared to traditional software testing.

H1 █▃▁▁▁▁ receives an average score of 4.68, with 94% of the participants agreeing/strongly agreeing with this best practice.

- 👍*Representative and sufficient data are generalized. It can avoid bias as much as possible, and improve accuracy, robustness, and fairness.*
- 👎*Good algorithm may compensate the effect of lack of data.*

H2 ▇▂▂▁▁▁ receives an average score of 4.48, with 84% of the participants agreeing/strongly agreeing with this best practice.

- 👍*AI can work with large data sets and perform complex calculations.*
- 👍*Computing power is proportional to user experience.*
- 👎*There exists many AI models that can run on low-end devices.*

H3 ▃▃▃▂▁▁ receives an average score of 3.82, we notice that there are 14% of the participants disagree/strongly disagree with this best practice, making it fail to be an acknowledged best practice. The point they are arguing is that after a well-prepared testing procedure, the testers do not need a high level of expertise, but can just follow the testing procedure and treat the AI system as a black box.

- 👍*High level of expertise for AI's algorithm is good for understanding model behavior, designing effective tests, interpreting results, debugging and troubleshooting.*
- 👎*If the testing procedure is well-prepared by experts, then the testers do not need a high level of expertise.*
- 👎*AI testing can be a black box where the tester may know nothing about the implementation.*

4.3.2 ***Waterfall & Agile (H4-H5).*** These two best practices discuss the differing workflows in AI system development, contrasting production-level and proof-of-concept systems. We note that most interviewees, except P2, P4, P6, and P12, work on production-level AI system development, while the latter four focus on proof-of-concept systems development.

The development of a proof-of-concept system primarily concentrates on assessing the practicality of advanced AI technologies within a specific business context. The foremost objective in this stage is the creation of a functional prototype, which, notably, is not required to meet production-level standards. Conversely, the development of a production-level system is dedicated to the construction of a high-quality AI system. This system is characterized by its thorough vetting through comprehensive QA procedures, ensuring its readiness for deployment in a production environment.

These two groups employ distinct workflows: the proof-of-concept team favors the waterfall workflow, valuing its linear, structured approach for clear, well-defined tasks. For example, P2 remarks, *"In the proof of concept stage, the waterfall method...offers a clear structure to follow, making it suitable for straightforward tasks."* Conversely, interviewees working on production-level AI systems adopt the agile workflow, with P1 and P3 specifically using Scrum. Agile workflow is preferred for its rapid iteration and flexibility, crucial in the uncertain terrain of AI engineering. P11 encapsulates this mindset: *"AI is full of uncertainty...fail fast, fail cheaply."* P3 highlights the faster iteration cycles in the early stages, attributing this to the trial-and-error nature of AI engineering, which often requires adjustments based on unexpected findings in data or algorithmic performance.

H4 (waterfall workflow) ▂▂▄▂▂▂ receives an average score of 3.17, with 34% of the participants agreeing/strongly agreeing with this best practice and 26% of the participants disagreeing/strongly disagreeing with this best practice, making it fail to be an acknowledged best practice.

- 👍*Using the waterfall workflow can help to ensure the completeness of the prototype to be achieved quickly, minor details are abstracted as we only want to show that a certain technology is effective on the target environment.*

- 👎*The waterfall model may not be the best choice for a quick proof-of-concept validation of an AI system. Because it is lack of flexibility. And if problems are discovered during testing, they can be expensive and time-consuming to fix.*
- 👎*This needs to be determined on a case-by-case basis.*

H5 (agile workflow) ▬■▬___ receives an average score of 3.94, with 66% of the participants agreeing/strongly agreeing with this best practice. This best practice failed to be an acknowledged best practice, but only 6% of the participants disagree/strongly disagree with this best practice.

- 👍*Agile is important for AI software development whereas time is important in occupying the market.*
- 👍*Agile workflow is good for dealing with uncertainty.*
- 👎*The best methodology depends on the specific needs and circumstances of the project.*

We note that H4 & H5 have a high neutral (3) rate of 32% and 22% respectively, feedback from the participants explains that most of the participants rate neutral (3) because they are not sure about the particular workflow their companies are adopting, while the rest think that the best workflow depends on the specific needs and circumstances of the project.

4.3.3 ***QA4AI knowledge source (H6-H7).*** Investigating the source of knowledge in QA4AI, we explore whether it is predominantly derived from accessible resources like literature and online platforms, or practical experience. Interviewees P1, P4, P5, and P6 highlight the importance of practical experience due to QA4AI's domain-specific nature, while P3 and P10 highlight the comprehensiveness of online resources which they find comprehensive due to the internet's convenience. P9 points out the necessity of both, with online resources offering knowledge access and practical experience providing application skills. This leads to a reclassification of knowledge types: *conceptual knowledge* from accessible resources and *business-specific knowledge* from practical experience. We conclude that conceptual knowledge forms the QA4AI foundation, and business-specific knowledge is crucial for real-world application.

H6 (conceptual knowledge) ■■▬___ receives an average score of 4.31, with 86% of the participants agreeing/strongly agreeing with this best practice.

- 👍*There are tons of excellent resources online to help you learn any conceptual knowledge you are not familiar with.*
- 👍*Nowadays, an excellent AI engineer should have the ability to learn new knowledge online quickly.*
- 👍*Many new concepts are coming out every day, and without the help of accessible resources, it is hard to keep up with the trend.*

H7 (business-specific knowledge) ■■▬___ receives an average score of 4.31, with 84% of the participants agreeing/strongly agreeing with this best practice.

- 👍*Practice makes perfect.*
- 👍*Business-specific knowledge is important for applying conceptual knowledge to real-world scenarios, and it is hard to obtain from accessible resources.*

Respondents reach a consensus on our classification of knowledge types and corresponding primary sources, with only 2% and 4% of the participants disagreeing/strongly disagreeing with H6 and H7 respectively.

4.3.4 ***Dataset source (H8-H9).*** Many types of data sources can be used to train and evaluate AI models, such as open-source datasets, self-collected datasets, and AI-generated datasets. In these two best practices, we explore the decision-making process for selecting data sources in AI model training and evaluation, focusing on two primary scenarios: proof-of-concept and production-level systems. P1 mentions that for proof-of-concept AI systems, open-source datasets are predominantly favored due to their accessibility, comprehensive annotation, and community validation. These datasets, often accompanied by benchmark results, facilitate easy performance comparison across various models.

Conversely, in the context of production-level AI systems, the preference shifts towards self-collected datasets. P6 mentions that these datasets are preferred for their relevance and representativeness in specific business scenarios, particularly in niche domains where open-source datasets may not align perfectly. Despite their advantages, self-collected datasets pose challenges in terms of the time and cost required for human annotation. Consequently, open-source datasets that closely mirror business scenarios are sometimes considered alternatives. Interestingly, despite studies indicating the high quality of AI-generated datasets [130], our interviewees express reservations about employing them in production, primarily due to trust issues with these data sources.

H8 (open-source datasets) ■▬▬___ receives an average score of 4.18, with 74% of the participants agreeing/strongly agreeing with this best practice.

- 👍*A common ache of AI models is low generalizability, so testing on diverse datasets is important. The variability of open-source datasets is a good way to test the generalizability of AI models.*
- 👍*Open-source datasets are usually validated by the community, with many benchmark results available, facilitating easy performance comparison across various models.*
- 👎*Open-source datasets are often used for proof-of-concept (PoC) projects in AI. However, they may not fully represent the complexity and variability of real-world data.*

H9 (self-collected datasets) ▬■■▬__ receives an average score of 3.45, with 50% of the participants agreeing/strongly agreeing with this best practice. We notice that there are 30% of the participants who vote neutral (3) and 18% of the participants disagree/strongly disagree with this best practice, making it fail to be an acknowledged best practice. The main reason they explain is that self-collected datasets will be very expensive and time-consuming when training a large model that requires a large amount of data, and many open-source datasets are close to their business scenarios with relatively high quality.

- 👍*Train with the data you'll use in production; the model you craft will rock solidly.*
- 👎*Self-collected datasets are usually more expensive and time-consuming to obtain. Sometimes open-source datasets are good enough.*

4.3.5 ***Data annotation (H10)****.*

As highlighted in sections H8 & H9, the process of data annotation is both time-intensive and expensive. Humans, prone to errors, may introduce inconsistencies in annotation, often varying between different annotators as mentioned by P3. A common practice for maintaining annotation quality, detailed by P6, involves employing multiple annotators for the same dataset to minimize errors. However, due to cost and time limitations, this is not always practical. P10 proposes an efficient data annotation method where data is annotated by humans once and initially validated by an operational model. Human intervention occurs only when discrepancies arise between model and annotator outputs, making this approach effective for frequently updated, high-accuracy models.

H10 ▬■■__▬ receives an average score of 3.85, with 60% of the participants agreeing/strongly agreeing with this best practice. It fails to be an acknowledged best practice because there are 32% of the participants who vote neutral (3), they mention that they are not sure about the effectiveness of this practice because it is not adopted in their companies.

- 👍*An operational model that is frequently updated can quickly and cheaply process large amounts of data, reducing the need for time-consuming and costly human validation.*

- 👎*This trick is only feasible when the model is already in production with relatively high accuracy, and newly labeled data is needed for frequent model retraining.*

4.3.6 ***Model update (H11-H12).*** H10 introduces a strategy to maintain annotation quality during model retraining, leading to discussions on model updates. Data drift, a prevalent issue in AI systems, occurs when the real-world data distribution diverges from the training set's distribution at deployment. Several interviewees (P1, P2, P3, P6, P8, P10, P11) acknowledge data drift in their products, necessitating frequent model updates to preserve performance. They suggest various triggers to perform model updates, which motivates H11:

- **Periodic updates:** Depending on the business context, models are updated at regular intervals, such as seasonally for P3 (fraud detection) or weekly for P8 (recommender system).
- **Performance-driven updates:** Monitored metrics like accuracy and F1 score prompt updates when they fall below a predefined threshold (P3, P6).
- **Data-driven updates:** Accumulation of a significant amount of labeled data triggers an update (P10).

P1 highlights a critical scenario: when the performance indicator drops significantly on the currently deployed model, and retraining using the latest data does not help, it may indicate that the model architecture is not suitable for the current business scenario, necessitating a redesign. This leads to H12, which explores the triggers for model redesigns.

H11 ▬■▬▁▁▁ receives an average score of 4.06, with 78% of the participants agreeing/strongly agreeing with this best practice.

- 👍*These three triggers are in line with my project's model update strategy.*
- 👎*it is also important to monitor the data itself for signs of drift and to understand the underlying causes of any changes in the data. In addition, while regular updates can help maintain model performance, they also come with computational costs.*

H12 ▬▬▬▬▁▁ receives an average score of 3.55, with 54% of the participants agreeing/strongly agreeing with this best practice. There are 22% of the participants who vote neutral (3) and 22% of the participants disagree/strongly disagree with this best practice, making it fail to be an acknowledged best practice. The opponents mainly argue about other possibilities that cause the performance drop and retraining to be ineffective.

- 👍*As document counterfeiting methods advance, the existing model architectures may become less effective in fraud detection. Then retraining doesn't help, thus exploring new model architectures becomes essential.*
- 👎*Need to check if there's any error in the data first. There might be a leak in the data pipeline that contaminates the latest data used in testing and retraining*

4.3.7 ***Data cleaning (H13-H14).*** Data cleaning, a crucial step for ensuring data and AI quality, involves various tasks such as removing duplicates, dropping outliers, and filling missing values. As gathered from interviewee responses, the primary motivations for data cleaning are twofold:

- **Enhancing Data Quality:** All interviewees acknowledge that data cleaning improves data quality, facilitating better learning of real-world patterns by AI models.
- **Ensuring Compliance:** Participants P5, P8, and P10 highlight the need for removing sensitive data, like Personal Identifiable Information (PII), to comply with data protection laws.

Many interviewees (P1, P2, P3, P5, P8, P10) emphasize the necessity of automating this process. Automated cleaning efficiently filters and formats data, leaving only the outliers for manual cleaning, thus optimizing cost-efficiency. It motivates H13 which discusses automated data cleaning using filter constraints. For example, when collecting date data from two different sources, the date format may be different, and the data cleaning script can automatically detect the date format and convert it to a unified format. Manual check and cleaning on the entire dataset is not feasible due to the large amount of data, therefore, H14 suggests that manual check and cleaning is only needed for outliers dropped by data cleaning scripts.

H13 ▅▄▃▁▁▁ receives an average score of 4.12, with 76% of the participants agreeing/strongly agreeing with this best practice.

- 👍*Initial data cleaning using filters is always step one in my data cleaning process.*
- 👍*Data cleaning is a time-consuming and costly process, automation using filters is needed to ensure the efficiency of data cleaning.*

H14 ▃▅▄▄▁▁ receives an average score of 3.47, with 56% of the participants agreeing/strongly agreeing with this best practice. We find that 40% of the participants vote neutral (3) and 38% of the participants disagree/strongly disagree with this best practice, making it fail to be an acknowledged best practice. The disagreements are mainly highlighting the complexity of real-world data and the difficulty of automating data cleaning. They believe that many cases cannot be handled by data cleaning scripts, thus manual check and cleaning is needed for the entire dataset. Some respondents also highlight the case that the data that passes the data cleaning scripts is sufficient to train the model, therefore manual check might not be needed, and outliers can be dropped directly.

- 👍*Human intervention is costly, only use when necessary.*
- 👎*Automated data cleaning methods can handle many common issues, but they may not be able to address all problems, especially those that require domain knowledge or complex decision-making.*
- 👎*Sometimes the qualified data is sufficient to train the model, and manual check is not needed.*

4.3.8 *Model choice upon different data types (H15-H16).*

Categorized by data types, there are mainly three types of data, structured data, unstructured data and semi-structured data. Structured data exists in fixed fields within records, like in relational databases. Unstructured data lacks a specific format and includes text, audio, video, etc. Semi-structured data, while not in relational databases, has some organizational properties for easier analysis. Our interviewees don't focus on semi-structured data, so we concentrate on structured and unstructured data.

Interviewees (P3, P6, P8, P11, P12) observe that modeling structured data is closely tied to business contexts, making it challenging to directly adopt open-source models. Also, pre-trained models for structured data are rare, suggesting a need for custom model design. However, this doesn't mean starting entirely from scratch; typically, a demo code with the desired architecture is used as a template.

Conversely, transfer learning is prevalent in unstructured data due to the transferability of abstracting high-level features across domains. Interviewees (P1, P2, P8, P10, P13) note the trend of large pre-trained models in AI, offering diverse sizes, datasets, and architectures. They concur that for difficult tasks, it is advisable to start with a pre-trained model and meticulously fine-tune it with domain-specific data.

These discussions motivate H15 and H16, which explore primary model choice for structured and unstructured data respectively.

H15 ▄▄▄▁▁▃ receives an average score of 3.91, with 56% of the participants agreeing/strongly agreeing with this best practice. We note that there are 28% of the participants vote neutral (3) and 16% of the participants disagree/strongly disagree with this best practice, making it fail to be an acknowledged best practice.

- 👍*Structured data is closely tied to business scenarios, therefore it is hard to fetch a model from open source and use it directly.*

- 👎*It depends on the specific problem and the amount of data you have.*

H16 ▃▅▅▁▁▁ receives an average score of 3.84, with 24% of the participants agreeing/strongly agreeing with this best practice. 28% of the participants vote neutral (3), and 16% disagree or strongly disagree with this best practice, causing it to fail to be recognized as a best practice.

- 👍*By fine-tuning these models on your specific task, you can leverage these learned features, which can lead to better performance than training a model from scratch, especially when you have limited data.*
- 👎*Fine tune does not always derive performance gain.*

4.3.9 ***Model performance bare minimum (H17).*** This best practice explores the lowest acceptable bounds of an AI model. Initially, we hypothesize a domain-specific, experience-based minimum AI performance threshold. However, insights from interviews with industry professionals (P1, P3, P6, P10) shift our focus. They highlight that for business-oriented AI projects, the paramount factor is not performance alone, but its business impact.

Particularly, P3 underlines that an AI model's minimum acceptable performance should derive from its business implications. For example, in anti-money laundering scenarios, the false positive rate, as noted by P3, correlates with financial losses. Thus, a formula exists to determine this rate's minimum threshold based on potential financial losses. Similarly, P6 points out the direct link between a recommender system's latency, user experience, and retention rates, leading to a formula for calculating minimum latency based on user retention.

Consequently, quantifying business impact is crucial. This involves identifying key performance indicators, continually monitoring them, and establishing their minimum thresholds in relation to the business impact. This motivates H17 to discuss the mapping between business impact and AI performance.

H17 ▃▅▅▁▁▁ receives an average score of 3.84, with 63% of the participants agreeing/strongly agreeing with this best practice. 34% of the participants vote neutral (3), making it fail to be an acknowledged best practice. Respondents who vote neutral (3) mention that they aren't involved in the business impact analysis process, and they are not sure about the mapping between business impact and AI performance.

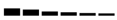
- 👍*The ultimate goal of most AI systems is to drive business value, and the technical performance of a model cannot directly reflect business impact.*
- 👎*It is hard to quantify the business impact.*

4.3.10 ***Focus transition before and after deployment (H18).*** In our QA process analysis, we observe a focus shift surrounding model deployment. Initially, while testing various models and techniques, the emphasis is predominantly on model performance, with the business impact being less defined, as indicated by P1, P3, and P8. Post-deployment, as more live data flows in, the business impact becomes clearer, shifting the focus to business outcomes, as noted by P1, P3, P8, and P9. Examples of business performance metrics include user retention rates, customer rates, and profit.

H18 ▅▅▃▁▁▁ receives an average score of 4.17, with 80% of the participants agreeing/strongly agreeing with this best practice.

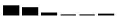
- 👍*While a model might perform well according to technical metrics, it is the impact on business-performance metrics that determines its real-world value.*
- 👍*AI systems are commercial products, therefore business metrics are important indicators after model deployment to evaluate the quality of them.*

4.3.11 ***Connecting stakeholders (H19).*** The primary objective of software, including AI systems, is to meet stakeholder needs. P2 highlights the importance of engaging with developers to identify specific needs, such as pinpointing the location, category, and possible causes of code vulnerabilities for a vulnerability detection system. P3 underscores regular interactions with financial institutions, i.e., their clients, to align with their requirements and expectations. P3's company encourages developers to engage directly with clients to grasp their needs. This sharing motivates H19, which discusses the importance of interviewing/researching stakeholders to understand their needs.

H19  receives an average score of 3.66, with 58% of the participants agreeing/strongly agreeing with this best practice. 16% of the participants vote neutral (3) and 20% of the participants disagree/strongly disagree with this best practice, making it fail to be an acknowledged best practice. The opponents mainly argue that the stakeholders are not always right, and the AI engineers should have the ability to judge whether the stakeholders' needs are reasonable.

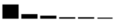
- 👍*Stop assuming what the stakeholders want, and start asking them.*
- 👎*Many requirements are not obtained from the stakeholders, but created from the air by the engineers. No one needed an iPhone before it was invented.*

4.3.12 ***Feedback training (H20).*** Inspired by ChatGPT's feedback button, P8 also mentions another way of connecting stakeholders - feedback training. *"We engineers are planning to directly collect the feedback of the AI-powered component of our product, and utilize the feedback as a discriminator to improve the AI quality."* (P8) This process makes the feedback collection efficient and quantified.

H20  receives an average score of 4.4, with 86% of the participants agreeing/strongly agreeing with this best practice.

- 👍*Feedback training can let the AI model listen to the end user's preference.*
- 👍*What a cost-efficient way to collect and make use of feedback!*

4.3.13 ***Trustworthy AI (H21).*** The unpredictable nature of AI challenges its trustworthiness. P3, P6, P8, and P10 mention *transparency* as a key factor in trustworthy AI. That is, we should let customers know why the AI makes certain decisions. P3 mentions that the interpretability work of their project is to ensure the transparency of their AI model, as mentioned in RQ1, clear statements of the model's decision-making process are important for the financial institutions and end users to understand what actions might be deemed as money laundering. P10, which is working as an intelligence provider, mentioned that they should let the end user know the websites the intelligence comes from. P8 mentions that they should let the end user know why the recommendation is made. For instance, one recommendation might be based on a user's order history over the past five days, while another could be due to the minimal distance between the item's location and the customer. The effort to make the AI's decision-making process transparent helps normal users understand the AI better and trust it more.

H21  receives an average score of 4.48, with 86% of the participants agreeing/strongly agreeing with this best practice.

- 👍*Transparency is the key to increase customer retention rate.*
- 👎*No matter how to explain a decision, it is still a black box.*

5 DISCUSSION

5.1 Suggestions to practitioners

5.1.1 ***Dos and don'ts for real-time AI systems.*** Many AI systems are deployed in real-time scenarios. The key challenge for real-time AI systems is a trade-off between *correctness* and *efficiency*, the latter of which includes latency and computational cost. It is crucial to recognize the paramount importance of achieving both low latency and cost-effectiveness in real-time AI systems. Therefore, during the initial development of real-time AI systems, blindly pursuing high accuracy by building up large models is not a good idea as later in the deployment phase, the AI system may not be able to meet the latency and cost requirements.

The demand for more compact models that maintain high levels of accuracy underscores the significance of diverse model compression techniques. A multitude of these techniques, originally proposed in academic circles, have found validation within the industry. For example, Fu et al. [43] and Bhardwaj et al. [24] discuss model compression for IoT applications. Consequently, industry practitioners should be cognizant of these model compression strategies and consider incorporating them into their artificial intelligence systems. Adjusting the hardware environment (e.g., CPU-only) and software environment (e.g., efficient model pipeline) are also effective ways to improve the efficiency of real-time AI systems that industry practitioners should consider.

5.1.2 ***Dos and don'ts towards data robustness***. Data *robustness* of AI systems is a big concern for domains with fast-changing data distribution. In such cases, the AI system may suffer from data drift [59, 82, 94], which is the phenomenon where the data distribution of the training data and the data distribution of the testing data are different. As H1 points out, an excellent AI model always requires representative and sufficient data, therefore, industry practitioners could collect data periodically and retrain the model using the latest data to mitigate data drift.

How often shall we collect data and retrain the model? H11 suggests that we should regularly update AI models using time intervals, performance metrics, and the availability of newly labeled data as triggers. These triggers can help to ensure the model's robustness efficiently.

However, one needs to take note that no model architecture can guarantee to be robust forever, data drift might make one model architecture ineffective in the new data distribution. Therefore, as H12 suggests, when performance drops and re-training does not help, one should consider redesigning the model architecture.

5.1.3 ***Effective utilization of open source resources***. The quality of open-source resources is recognized by both interview practitioners and survey participants, highlighting the importance of open-source resources for QA4AI. In our study, the following are several aspects that industry practitioners should consider to utilizing open-source resources.

Open-source datasets. H1 highlights the importance of data quality, thus obtaining high-quality data is the first step to ensure AI quality. However, collecting data is a time-consuming and labor-intensive process, therefore, industry practitioners should consider utilizing open-source datasets to save time and cost.

As H8 points out, open-source datasets are the primary source for proof-of-concept usage, in such cases the ultimate goal is to demonstrate the feasibility of a certain AI model, therefore, open-source datasets are sufficient. H9 suggests that self-collected data is more suitable for production-level AI systems, as the data is more representative and sufficient. However, it is not recognized by questionnaire participants, with many respondents highlighting the feasibility of using open-source datasets for production-level AI systems. Compared with self-collected data, open-source datasets are more cost-effective, and many popular open-source datasets are widely validated by the community, ensuring the quality of the data and also providing many benchmarks for practitioners to evaluate their AI models. What's more, particularly for large models that are data-hungry, open-source datasets are the only choice for many practitioners.

Therefore, we suggest that industry practitioners should consider utilizing open-source datasets for proof-of-concept usage and also for production-level AI systems, but they should be careful to choose the right open-source datasets that are domain-related.

Open-source models. Pre-trained models are also widely recognized by interview practitioners and survey participants, as they can save time and cost for practitioners. Similar to open-source datasets, open-source models are also widely validated by the community, ensuring the quality of the models.

H16 indicates a practice of utilizing a pre-trained model as a baseline model and then fine-tuning the model to fit the target task better when dealing with unstructured data. However, it is 0.16 away from being a well-supported practice, with respondents questioning the effectiveness of fine-tuning. They claim that fine-tuning is not always effective, and sometimes it may even hurt the performance of the model. Therefore, we suggest that industry practitioners should be careful when fine-tuning a pre-trained model, and try to select sufficient and representative data for fine-tuning in order to make real improvements.

Open-source learning materials. No one is a born QA4AI expert, therefore, learning materials are essential for practitioners to learn QA4AI knowledge and skills. As H6 indicates, open-source learning materials are the primary source for practitioners to learn conceptual QA4AI knowledge and skills. We provide many options for interview practitioners to choose from, such as online recourses and literature, company internal training, and courses, most of the interviewees choose online resources and literature. But they also point out that not all the QA4AI knowledge and skills can be learned from online resources and literature, as H7 suggests, business-specific skills can only be learned from hands-on experience.

Therefore, we suggest that industry practitioners should utilize open-source learning materials to learn conceptual QA4AI knowledge and skills, but hands-on experience is also essential for practitioners to learn business-specific skills.

5.2 Threats to Validity

Despite our best efforts to recruit interview practitioners as many as possible, the number of interviewees is still a small set (15) compared to the number of AI practitioners in the industry. This might cause a certain bias in our interview study, e.g., missing insights from a certain angle. However, the number of interviewees is sufficient to achieve data saturation as mentioned in Section 3. This is similar to many prior interview studies in the software engineering domain, such as [117] (14 interviewees), [35] (18 interviewees) and [78] (14 interviewees). We also particularly recruit interviewees from different roles and companies to ensure the diversity of our interviewees to mitigate this threat. Similarly, the number of survey respondents is also a small set (50) compared to the number of AI practitioners in the industry. But this number is also similar to many other survey studies in the software engineering domain, e.g., [78] (46 respondents), [69] (10 export respondents).

Although we design the interview questions based on plenty of literature, it is still possible that we might miss some important questions. We limit this threat by employing two predetermined criteria for conducting literature searches, i.e., 11 QA4AI properties and 9 stages of AI development, in order to ensure the completeness of our interview question design. For several roles, we only manage to interview one participant, giving us a one-sided perspective, therefore it may have a certain bias. We mitigate this threat by cross-checking the interview results with other interviewees and also with the survey results. Additionally, the ranking of QA4AI qualities is quite objective, we observe that some interviewees intend to give high grades while some intend to give low grades. This threat is mitigated by confirming the ranking again with the interviewees after they explain the reason behind their ranking, some of them indeed change their ranking after the confirmation as they realize that the explanation they give is not consistent with the ranking they give.

6 RELATED WORK

In this section, we discuss the related work in two aspects: (1) quality assurance for AI systems, and (2) AI development process analysis.

6.1 Quality Assurance for AI Systems

Due to the nondeterministic nature of AI, ensuring the quality of AI systems is challenging. Several studies concentrate on assessing the QA4AI across various dimensions. Murphy et al. [88] mentioned a software testing approach for ML applications and implemented it on two ML ranking algorithms: Support Vector Machines [58] and MartiRank [50]. Breck et al. [25] proposed 28 specific tests and monitoring needs summarized from the experience of ML system production, they also presented a road map to improve ML production readiness and pay back technical debt. Similarly, in our paper, the best practices we summarize also contain specific tests and monitoring needs. However, our work focuses on a wider scope, i.e., quality assurance for AI systems, instead of testing within a narrow definition. Ma et al. [83] proposed *DeepMutation*, a mutation testing framework designed for deep learning systems, which can measure the quality of test data. Ghadesi et al. [44] mined Stack Overflow by studying 11449 stack traces to identify the causes of exceptions in AI systems and provided insights for improving the quality of AI libraries and applications. Kim et al. [73] proposed a novel criterion called *Surprise Adequacy* to measure the adequacy of deep learning system's test data, demonstrating that utilizing the surprise adequacy criterion for sampling could enhance the classification precision of deep learning systems against adversarial instances by a maximum of 77.5%. Data adequacy is one of our interview questions. Many studies [102, 110] validated the effectiveness of data resampling methods for imbalanced data, recognizing that data resampling methods can improve the correctness of AI systems. There are many over-sampling methods aiming at increasing correctness under imbalanced data scenarios, such as *SMOTE* [30], *Borderline-SMOTE* [54], *ADASYN* [54], data argumentation [87, 116] and so on. Zhang et al. [136] proposed *Perturbed Model Validation*, a novel method to validate the model relevance of AI systems and detect overfitting/underfitting. Many model compression methods (e.g., pruning [106], quantization [66], knowledge distillation [62]) were proposed to improve the efficiency of AI systems. *Green AI* [107] is another trending topic that focuses on the energy efficiency of AI systems. This term emphasizes the importance of AI efficiency in order to decrease its carbon footprint. Rotman [104] discussed practices overcoming the deployability challenges of ML-powered networks. Hamon et al. [53] discussed the robustness and explainability (similar idea as interpretability) of AI systems and proposed policies to provide a regulatory framework for AI usage. *Explainable AI* (XAI) is another trending topic to ensure the quality of AI systems, XAI aims to explain the decision-making process of AI systems. Arrieta et al. [20] provided a comprehensive survey of XAI, including the definition, taxonomy, opportunities and challenges, providing newcomers with reference literature about future research directions. Li et al. [77] proposed *Named Entity Recognition* to ensure the privacy of AI systems. There are also many studies about differential privacy [15, 133, 134] that ensure the privacy of AI systems by adding noise to the training data. There are studies focused on the security and privacy of AI systems, such as training data extraction attacks [29, 65, 126], data poisoning attacks [16, 115, 131] and model inversion attack [41, 57, 138]. Song et al. [111] proposed a novel method to evaluate the transferability of AI systems. Response perturbation [70, 76] is a method to improve the security of AI systems by preventing model stealing attacks. There are also many researches [67, 85, 118] that focused on the fairness of AI systems. Our study covers all the 11 QA4AI properties above, aiming to reveal the importance, challenges and corresponding solutions of these properties under the industry context.

6.2 AI Development Process Analysis

There are studies about the current status, challenges and best practices of AI development. Amershi et al. [18] summarized a 9-stage workflow process for AI development, extracting challenges and best practices for each stage from interviews with Microsoft's AI developers. Our study follows the

9-stage workflow process and design interview questions that cover each stage. Serban et al. [108] mined literature and summarized 29 best practices for ML applications, then conducted a survey to observe the adoption level of these practices in the industry. Our study also summarizes best practices for AI development, but we focus on the practices of AI quality assurance. Fan et al. [36] summarized good practices to enhance the quality of open-source AI repositories, making them more popular in the community, by mining academic open-source AI repositories. Yang et al. [129] mined 24953 GitHub issues from open-source AI repositories, analyzed the metadata of these issues and summarized the best practices for resolving AI-related issues quickly. Paleyes et al. [93] compared flow-based programming with the current prevalent service-oriented paradigm, showing that flow-based programming is beneficial for discovering and collecting data for AI software. Song et al. [112] explored the potential of utilizing practices proposed by academic studies in the industry through an interactive rapid review with industry practitioners. Practitioners in our study also mentioned many techniques and tools proposed by academic studies and recognized their practical values. Cabrera et al. [26] surveyed real-world ML-powered systems from a Data-Oriented Architecture (DOA) perspective, discussing how DOA can aid ML deployment challenges and the adoption level of DOA in real-world ML systems. Wan et al. interviewed 14 people and surveyed 342 people to understand the differences between machine learning (AI) systems and non-machine learning system development [119]. They reported differences in the development process when machine learning is incorporated into the system.

7 CONCLUSION AND FUTURE WORK

Aimed to understand QA4AI under industry contexts, we conducted an interview study with 15 AI industry practitioners from various roles, countries, and company sizes. From the literature on QA4AI, we identified 11 QA4AI properties that could provide a holistic view of the quality of AI systems. Interviewees were asked to rank the importance of these properties and to identify the challenges and solutions for each property they encountered. Numerous other questions covering the whole AI development process were also posed to identify best practices for QA4AI. We summarized 21 QA4AI practices from the interviews with a survey involving 50 respondents. We discussed the importance of each QA4AI property and the reasons behind their rankings in RQ1. Findings were summarized for each property. We also found a set of challenges and solutions for overcoming each challenge for each property, detailed in RQ2. Finally, we discussed the best practices for QA4AI in RQ3. We identified that 10 practices were well-supported by practitioners and 8 practices were marginally agreed by the participants. For each of these practices, we discussed supporting and counter-evidence from the interviews. We also analyzed the reasons why some practices were not well-supported by practitioners. Our study provides an investigation into the key concerns, challenges, and best practices of QA4AI in an industry context. We provided insights into what industry practitioners should focus on when developing AI systems, potential obstacles they might encounter, and how to overcome them. We also identified 10 QA4AI practices that were well-supported and 8 practices that were marginally agreed by the participants, which could be used as a checklist to ensure the quality of AI systems.

Future work in QA4AI should focus on areas such as conducting a finer-grained analysis of industry-specific needs for tailored best practice guidelines, undertaking a longitudinal study to monitor the evolution of QA4AI challenges and practices, and developing innovative tools for automating evaluation processes in QA4AI.

REFERENCES

[1] [n. d.]. Apache Ignite. https://ignite.apache.org/
[2] [n. d.]. Apache Spark. https://spark.apache.org/

[3] [n. d.]. Kubernetes. https://kubernetes.io/

[4] [n. d.]. NVIDIA CUDA toolkit. https://developer.nvidia.com/cuda-toolkit

[5] [n. d.]. NVIDIA TensorRT. https://developer.nvidia.com/tensorrt

[6] [n. d.]. NVIDIA Triton Inference Server. https://developer.nvidia.com/nvidia-triton-inference-server

[7] [n. d.]. Personal Data Protection Act. https://www.pdpc.gov.sg/Overview-of-PDPA/The-Legislation/Personal-Data-Protection-Act

[8] [n. d.]. Pinecone. https://www.pinecone.io/

[9] [n. d.]. PyTorch. https://pytorch.org/

[10] [n. d.]. Seldon. https://www.seldon.io/

[11] [n. d.]. TensorFlow. https://www.tensorflow.org/

[12] 2014. History of the Basel Committee. https://www.bis.org/bcbs/history.htm

[13] 2015. ISO 9001:2015. https://www.iso.org/standard/62085.html

[14] 2022. General Data Protection Regulation (GDPR). https://gdpr-info.eu/

[15] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 308–318.

[16] Ibrahim M Ahmed and Manar Younis Kashmoola. 2021. Threats on machine learning technique by data poisoning attack: A survey. In *Advances in Cyber Security: Third International Conference, ACeS 2021, Penang, Malaysia, August 24–25, 2021, Revised Selected Papers 3*. Springer, 586–600.

[17] 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)*. 291–300. https://doi.org/10.1109/ICSE-SEIP.2019.00042

[18] 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)*. 291–300. https://doi.org/10.1109/ICSE-SEIP.2019.00042

[19] Shin Ando and Chun-Yuan Huang. 2017. Deep Over-sampling Framework for Classifying Imbalanced Data. arXiv:1704.07515 [cs.LG]

[20] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. 2019. Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI. arXiv:1910.10045 [cs.AI]

[21] Muhammad Hilmi Asyrofi, Zhou Yang, Imam Nur Bani Yusuf, Hong Jin Kang, Ferdian Thung, and David Lo. 2022. BiasFinder: Metamorphic Test Generation to Uncover Bias for Sentiment Analysis Systems. *IEEE Transactions on Software Engineering* 48, 12 (2022), 5087–5101. https://doi.org/10.1109/TSE.2021.3136169

[22] Yang Bao, Gilles Hilary, and Bin Ke. 2022. Artificial intelligence and fraud detection. *Innovative Technology at the Interface of Finance and Operations: Volume I* (2022), 223–247.

[23] Mohammad Riyaz Belgaum, Zainab Alansari, Shahrulniza Musa, Muhammad Mansoor Alam, and MS Mazliham. 2021. Role of artificial intelligence in cloud computing, IoT and SDN: Reliability and scalability issues. *International Journal of Electrical and Computer Engineering* 11, 5 (2021), 4458.

[24] Kartikeya Bhardwaj, Naveen Suda, and Radu Marculescu. 2019. Dream Distillation: A Data-Independent Model Compression Framework. arXiv:1905.07072 [stat.ML]

[25] Eric Breck, Shanqing Cai, Eric Nielsen, Michael Salib, and D. Sculley. 2017. The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction. In *Proceedings of IEEE Big Data*.

[26] Christian Cabrera, Andrei Paleyes, Pierre Thodoroff, and Neil D. Lawrence. 2023. Real-world Machine Learning Systems: A survey from a Data-Oriented Architecture Perspective. arXiv:2302.04810 [cs.SE]

[27] Longbing Cao. 2021. AI in Finance: Challenges, Techniques and Opportunities. arXiv:2107.09051 [q-fin.CP]

[28] Longbing Cao. 2022. AI in Finance: Challenges, Techniques, and Opportunities. *ACM Comput. Surv.* 55, 3, Article 64 (feb 2022), 38 pages. https://doi.org/10.1145/3502289

[29] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting Training Data from Large Language Models. arXiv:2012.07805 [cs.CR]

[30] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research* 16 (jun 2002), 321–357. https://doi.org/10.1613/jair.953

[31] Karel Crombecq, Luciano De Tommasi, Dirk Gorissen, and Tom Dhaene. 2009. A novel sequential design strategy for global surrogate modeling. In *Proceedings of the 2009 Winter Simulation Conference (WSC)*. 731–742. https:

//doi.org/10.1109/WSC.2009.5429687

[32] Daniela S. Cruzes and Tore Dyba. 2011. Recommended Steps for Thematic Synthesis in Software Engineering. In *Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement (ESEM '11)*. IEEE Computer Society, USA, 275–284. https://doi.org/10.1109/ESEM.2011.36

[33] Arun Das and Paul Rad. 2020. Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint arXiv:2006.11371* (2020).

[34] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[35] Omar Elazhary, Colin Werner, Ze Shi Li, Derek Lowlind, Neil A. Ernst, and Margaret-Anne Storey. 2022. Uncovering the Benefits and Challenges of Continuous Integration Practices. *IEEE Transactions on Software Engineering* 48, 7 (July 2022), 2570–2583. https://doi.org/10.1109/tse.2021.3064953

[36] Yuanrui Fan, Xin Xia, David Lo, Ahmed E Hassan, and Shanping Li. 2021. What makes a popular academic AI repository? *Empirical Software Engineering* 26, 1 (2021), 1–35.

[37] Michael Felderer and Rudolf Ramler. 2021. Quality Assurance for AI-Based Systems: Overview and Challenges (Introduction to Interactive Session). In *Software Quality: Future Perspectives on Software Engineering Quality*. Springer International Publishing, 33–42. https://doi.org/10.1007/978-3-030-65854-0_3

[38] 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* (Virtual Event, USA) *(ISSTA 2020)*. Association for Computing Machinery, New York, NY, USA, 177–188. https://doi.org/10.1145/3395363.3397357

[39] Stefan Feuerriegel, Mateusz Dolata, and Gerhard Schwabe. 2020. Fair AI: Challenges and opportunities. *Business & information systems engineering* 62 (2020), 379–384.

[40] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. 2019. All Models are Wrong, but Many are Useful: Learning a Variable's Importance by Studying an Entire Class of Prediction Models Simultaneously. arXiv:1801.01489 [stat.ME]

[41] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 1322–1333.

[42] Jerome Friedman. 2000. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics* 29 (11 2000). https://doi.org/10.1214/aos/1013203451

[43] Shipeng Fu, Zhen Li, Kai Liu, Sadia Din, Muhammad Imran, and Xiaomin Yang. 2020. Model Compression for IoT Applications in Industry 4.0 via Multiscale Knowledge Transfer. *IEEE Transactions on Industrial Informatics* 16, 9 (2020), 6013–6022. https://doi.org/10.1109/TII.2019.2953106

[44] Amin Ghadesi, Maxime Lamothe, and Heng Li. 2023. What Causes Exceptions in Machine Learning Applications? Mining Machine Learning-Related Stack Traces on Stack Overflow. arXiv:2304.12857 [cs.LG]

[45] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. 2014. Peeking Inside the Black Box: Visualizing Statistical Learning with Plots of Individual Conditional Expectation. arXiv:1309.6392 [stat.AP]

[46] Chen Gong, Zhou Yang, Yunpeng Bai, Jieke Shi, Arunesh Sinha, Bowen Xu, David Lo, Xinwen Hou, and Guoliang Fan. 2022. Curiosity-Driven and Victim-Aware Adversarial Policies. In *Proceedings of the 38th Annual Computer Security Applications Conference* (Austin, TX, USA) *(ACSAC '22)*. Association for Computing Machinery, New York, NY, USA, 186–200. https://doi.org/10.1145/3564625.3564636

[47] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. arXiv:1406.2661 [stat.ML]

[48] Leo Goodman. 1961. Snowball Sampling. *Ann Math Stat* 32 (03 1961). https://doi.org/10.1214/aoms/1177705148

[49] Serge Gorbunov and Arnold Rosenbloom. 2010. Autofuzz: Automated network protocol fuzzing framework. *Ijcsns* 10, 8 (2010), 239.

[50] Philip Gross, Albert Boulanger, Marta Arias, David L. Waltz, Philip M. Long, Charles Lawson, Roger Anderson, Matthew Koenig, Mark Mastrocinque, William Fairechio, John A. Johnson, Serena Lee, Frank Doherty, and Arthur Kressner. 2006. Predicting Electricity Distribution Feeder Failures Using Machine Learning Susceptibility Analysis. In *IAAI*. http://www.phillong.info/publications/GBAetal06_susc.pdf

[51] Greg Guest, Arwen Bunce, and Laura Johnson. 2006. How Many Interviews Are Enough?: An Experiment with Data Saturation and Variability. *Field Methods* 18, 1 (Feb. 2006), 59–82. https://doi.org/10.1177/1525822X05279903 Publisher: SAGE Publications Inc.

[52] Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. 2018. Dynamic Task Prioritization for Multitask Learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[53] Ronan Hamon, Henrik Junklewitz, Ignacio Sanchez, et al. 2020. Robustness and explainability of artificial intelligence. *Publications Office of the European Union* 207 (2020).

[54] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. 2005. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In *Proceedings of the 2005 International Conference on Advances in Intelligent Computing - Volume Part I* (Hefei, China) *(ICIC'05)*. Springer-Verlag, Berlin, Heidelberg, 878–887. https://doi.org/10.1007/11538059_91

[55] Miriam Harris, Amy Qi, Luke Jeagal, Nazi Torabi, Dick Menzies, Alexei Korobitsyn, Madhukar Pai, Ruvandhi R Nathavitharana, and Faiz Ahmad Khan. 2019. A systematic review of the diagnostic accuracy of artificial intelligence-based computer programs to analyze chest x-rays for pulmonary tuberculosis. *PloS one* 14, 9 (2019), e0221339.

[56] Mardhiya Hayati, Siti Mutmainah, and Syed Ghufran. 2021. Random and Synthetic Over-Sampling Approach to Resolve Data Imbalance in Classification. *International Journal of Artificial Intelligence Research* 4 (01 2021), 86. https://doi.org/10.29099/ijair.v4i2.152

[57] Zecheng He, Tianwei Zhang, and Ruby B Lee. 2019. Model inversion attacks against collaborative inference. In *Proceedings of the 35th Annual Computer Security Applications Conference*. 148–162.

[58] M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their Applications* 13, 4 (1998), 18–28. https://doi.org/10.1109/5254.708428

[59] Lukas Heiland, Marius Hauser, and Justus Bogner. 2023. Design Patterns for AI-based Systems: A Multivocal Literature Review and Pattern Repository. arXiv:2303.13173 [cs.SE]

[60] Henrik Heymann, Hendrik Mende, Maik Frye, and Robert H. Schmitt. 2023. Assessment Framework for Deployability of Machine Learning Models in Production. *Procedia CIRP* 118 (2023), 32–37. https://doi.org/10.1016/j.procir.2023.06.007 16th CIRP Conference on Intelligent Computation in Manufacturing Engineering.

[61] Hans-Martin Heyn, Eric Knauss, Amna Pir Muhammad, Olof Eriksson, Jennifer Linder, Padmini Subbiah, Shameer Kumar Pradhan, and Sagar Tungal. 2021. Requirement Engineering Challenges for AI-intense Systems Development. In *2021 IEEE/ACM 1st Workshop on AI Engineering - Software Engineering for AI (WAIN)*. 89–96. https://doi.org/10.1109/WAIN52551.2021.00020

[62] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. arXiv:1503.02531 [stat.ML]

[63] Carrie Howell, Wei Su, Ariann Nassel, April Agne, and Andrea Cherrington. 2020. Area based stratified random sampling using geospatial technology in a community-based survey. *BMC Public Health* 20 (11 2020). https://doi.org/10.1186/s12889-020-09793-0

[64] Krystal Hu. 2023. CHATGPT sets record for fastest-growing user base - analyst note. https://www.reuters.com/technology/chatgpt-sets-record-fastest-growing-user-base-analyst-note-2023-02-01/

[65] Shotaro Ishihara. 2023. Training Data Extraction From Pre-trained Language Models: A Survey. arXiv:2305.16157 [cs.CL]

[66] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2017. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. arXiv:1712.05877 [cs.LG]

[67] Jean-Marie John-Mathews, Dominique Cardon, and Christine Balagué. 2022. From reality to world. A critical perspective on AI fairness. *Journal of Business Ethics* 178, 4 (July 2022), 945–959. https://doi.org/10.1007/s10551-022-05055-8 FNEGE 1, HCERES A, ABS 3.

[68] Milan Jovic, Andrea Adamoli, and Matthias Hauswirth. 2011. Catch me if you can: performance bug detection in the wild. In *Proceedings of the 2011 ACM international conference on Object oriented programming systems languages and applications*. 155–170.

[69] Reza karemi and mohammadreza nasiri. 2023. Identifying and Prioritizing Factors Affecting Knowledge Sharing in Software Companies. *Sciences and Techniques of Information Management* (2023), –. https://doi.org/10.22091/stim.2023.10146.2043

[70] Sanjay Kariyappa and Moinuddin K Qureshi. 2019. Defending Against Model Stealing Attacks with Adaptive Misinformation. arXiv:1911.07100 [stat.ML]

[71] Salwa Khalil. 2014. Not everything that counts can be counted and not everything that can be counted counts. *The Psychiatric Bulletin* 38, 2 (April 2014), 86–86. https://doi.org/10.1192/pb.38.2.86b

[72] Jinhan Kim, Robert Feldt, and Shin Yoo. 2019. Guiding Deep Learning System Testing Using Surprise Adequacy. In *Proceedings of the 41st International Conference on Software Engineering* (Montreal, Quebec, Canada) *(ICSE '19)*. IEEE Press, 1039–1049. https://doi.org/10.1109/ICSE.2019.00108

[73] Jinhan Kim, Robert Feldt, and Shin Yoo. 2019. Guiding Deep Learning System Testing Using Surprise Adequacy. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE. https://doi.org/10.1109/icse.2019.00108

[74] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment Anything. arXiv:2304.02643 [cs.CV]

[75] Pavneet Singh Kochhar, Xin Xia, and David Lo. 2019. Practitioners' Views on Good Software Testing Practices. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*.

61–70. https://doi.org/10.1109/ICSE-SEIP.2019.00015

[76] Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. 2018. Defending Against Machine Learning Model Stealing Attacks Using Deceptive Perturbations. arXiv:1806.00054 [cs.LG]

[77] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2022. A Survey on Deep Learning for Named Entity Recognition. *IEEE Transactions on Knowledge and Data Engineering* 34, 1 (jan 2022), 50–70. https://doi.org/10.1109/tkde.2020.2981314

[78] Jenny T. Liang, Maryam Arab, Minhyuk Ko, Amy J. Ko, and Thomas D. LaToza. 2023. A Qualitative Study on the Implementation Design Decisions of Developers. arXiv:2301.09789 [cs.SE]

[79] Bowen Liu, Boao Xiao, Xutong Jiang, Siyuan Cen, Xin He, Wanchun Dou, and Huaming Chen. 2023. Adversarial Attacks on Large Language Model-Based System and Mitigating Strategies: A Case Study on ChatGPT. *Sec. and Commun. Netw.* 2023 (jan 2023), 10 pages. https://doi.org/10.1155/2023/8691095

[80] Qinghua Lu, Liming Zhu, Xiwei Xu, Jon Whittle, Didar Zowghi, and Aurelie Jacquet. 2023. Responsible AI Pattern Catalogue: A Collection of Best Practices for AI Governance and Engineering. *ACM Comput. Surv.* (oct 2023). https://doi.org/10.1145/3626234 Just Accepted.

[81] Scott Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. arXiv:1705.07874 [cs.AI]

[82] 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 *Agile Processes in Software Engineering and Extreme Programming*, Philippe Kruchten, Steven Fraser, and François Coallier (Eds.). Springer International Publishing, Cham, 227–243.

[83] Lei Ma, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Felix Juefei-Xu, Chao Xie, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. 2018. DeepMutation: Mutation Testing of Deep Learning Systems. arXiv:1805.05206 [cs.SE]

[84] 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 (apr 2022), 59 pages. https://doi.org/10.1145/3487043

[85] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2022. A Survey on Bias and Fairness in Machine Learning. arXiv:1908.09635 [cs.LG]

[86] Marvin Minsky. 1961. Steps toward Artificial Intelligence. *Proceedings of the IRE* 49, 1 (1961), 8–30. https://doi.org/10.1109/JRPROC.1961.287775

[87] Alhassan Mumuni and Fuseini Mumuni. 2022. Data augmentation: A comprehensive survey of modern approaches. *Array* 16 (2022), 100258. https://doi.org/10.1016/j.array.2022.100258

[88] Chris Murphy, Gail Kaiser, and Marta Arias. 2007. An Approach to Software Testing of Machine Learning Applications. 167–.

[89] Nadia Nahar, Shurui Zhou, Grace Lewis, and Christian Kästner. 2022. Collaboration Challenges in Building ML-Enabled Systems: Communication, Documentation, Engineering, and Process. arXiv:2110.10234 [cs.SE]

[90] Bret Nestor, Matthew BA McDermott, Willie Boag, Gabriela Berner, Tristan Naumann, Michael C Hughes, Anna Goldenberg, and Marzyeh Ghassemi. 2019. Feature robustness in non-stationary health records: caveats to deployable model performance in common clinical machine learning tasks. In *Machine Learning for Healthcare Conference*. PMLR, 381–405.

[91] Cuong Nguyen, Thanh-Toan Do, and Gustavo Carneiro. 2023. Task Weighting in Meta-learning with Trajectory Optimisation. arXiv:2301.01400 [cs.LG]

[92] Matthias Niedermaier, Florian Fischer, and Alexander von Bodisco. 2017. PropFuzz—An IT-security fuzzing framework for proprietary ICS protocols. In *2017 International conference on applied electronics (AE)*. IEEE, 1–4.

[93] Andrei Paleyes, Christian Cabrera, and Neil D Lawrence. 2021. Towards better data discovery and collection with flow-based programming. *arXiv preprint arXiv:2108.04105* (2021).

[94] Andrei Paleyes, Raoul-Gabriel Urma, and Neil D. Lawrence. 2022. Challenges in Deploying Machine Learning: A Survey of Case Studies. *ACM Comput. Surv.* 55, 6, Article 114 (dec 2022), 29 pages. https://doi.org/10.1145/3533378

[95] Worrawut Pananurak, Somphong Thanok, and Manukid Parnichkun. 2009. Adaptive cruise control for an intelligent vehicle. In *2008 IEEE International Conference on Robotics and Biomimetics*. 1794–1799. https://doi.org/10.1109/ROBIO.2009.4913274

[96] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, 311–318. https://doi.org/10.3115/1073083.1073135

[97] Riccardo Perego, Antonio Candelieri, Francesco Archetti, and Danilo Pau. 2020. Tuning deep neural network's hyperparameters constrained to deployability on tiny systems. In *International Conference on Artificial Neural Networks*. Springer, 92–103.

[98] Michael Pradel and Koushik Sen. 2018. Deepbugs: A learning approach to name-based bug detection. *Proceedings of the ACM on Programming Languages* 2, OOPSLA (2018), 1–25.

[99] Mohammad Rizky Pratama and Dana Sulistiyo Kusumo. 2021. Implementation of continuous integration and continuous delivery (ci/cd) on automatic performance testing. In *2021 9th International Conference on Information and Communication Technology (ICoICT)*. IEEE, 230–235.

[100] The Associated Press. 2022. Nearly 400 car crashes in 11 months involved Automated Tech, companies tell regulators. https://www.npr.org/2022/06/15/1105252793/nearly-400-car-crashes-in-11-months-involved-automated-tech-companies-tell-regul

[101] Sai Sathiesh Rajan, Sakshi Udeshi, and Sudipta Chattopadhyay. 2022. AequeVox: Automated Fairness Testing of Speech Recognition Systems. In *Fundamental Approaches to Software Engineering*, Einar Broch Johnsen and Manuel Wimmer (Eds.). Springer International Publishing, 245–267.

[102] Satyendra Singh Rawat and Amit Kumar Mishra. 2022. Review of Methods for Handling Class-Imbalanced in Classification Problems. arXiv:2211.05456 [cs.LG]

[103] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. arXiv:1602.04938 [cs.LG]

[104] Noga H. Rotman. 2023. Tackling Deployability Challenges in ML-Powered Networks. *SIGMETRICS Perform. Eval. Rev.* 51, 2 (oct 2023), 94–96. https://doi.org/10.1145/3626570.3626605

[105] Rebecca L. Russell, Louis Kim, Lei H. Hamilton, Tomo Lazovich, Jacob A. Harer, Onur Ozdemir, Paul M. Ellingwood, and Marc W. McConley. 2018. Automated Vulnerability Detection in Source Code Using Deep Representation Learning. arXiv:1807.04320 [cs.LG]

[106] Abdullah Salama, Oleksiy Ostapenko, Tassilo Klein, and Moin Nabi. 2019. Pruning at a Glance: Global Neural Pruning for Model Compression. arXiv:1912.00200 [cs.CV]

[107] Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2020. Green AI. *Commun. ACM* 63, 12 (nov 2020), 54–63. https://doi.org/10.1145/3381831

[108] 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)* (Bari, Italy) *(ESEM '20)*. Association for Computing Machinery, New York, NY, USA, Article 3, 12 pages. https://doi.org/10.1145/3382494.3410681

[109] Jiawen Shi, Yixin Liu, Pan Zhou, and Lichao Sun. 2023. BadGPT: Exploring Security Vulnerabilities of ChatGPT via Backdoor Attacks to InstructGPT. arXiv:2304.12298 [cs.CR]

[110] Jiang-Xin Shi, Tong Wei, Yuke Xiang, and Yu-Feng Li. 2023. How Re-sampling Helps for Long-Tail Learning? arXiv:2310.18236 [cs.CV]

[111] Jie Song, Yixin Chen, Xinchao Wang, Chengchao Shen, and Mingli Song. 2019. Deep model transferability from attribution maps. *Advances in Neural Information Processing Systems* 32 (2019).

[112] Qunying Song, Markus Borg, Emelie Engström, Håkan Ardö, and Sergio Rico. 2022. Exploring ML Testing in Practice: Lessons Learned from an Interactive Rapid Review with Axis Communications. In *Proceedings of the 1st International Conference on AI Engineering: Software Engineering for AI* (Pittsburgh, Pennsylvania) *(CAIN '22)*. Association for Computing Machinery, New York, NY, USA, 10–21. https://doi.org/10.1145/3522664.3528596

[113] Qunying Song, Markus Borg, Emelie Engström, Håkan Ardö, and Sergio Rico. 2022. Exploring ML testing in practice – Lessons learned from an interactive rapid review with Axis Communications. arXiv:2203.16225 [cs.SE]

[114] Anselm Strauss and Juliet M. Corbin. 1990. *Basics of qualitative research: Grounded theory procedures and techniques.* Sage Publications, Inc, Thousand Oaks, CA, US. Pages: 270.

[115] Gan Sun, Yang Cong, Jiahua Dong, Qiang Wang, Lingjuan Lyu, and Ji Liu. 2021. Data poisoning attacks on federated machine learning. *IEEE Internet of Things Journal* 9, 13 (2021), 11365–11375.

[116] Praveen Singh Thakur, Mahipal Jadeja, and Satyendra Singh Chouhan. 2022. Text Augmentation based Imbalance Learning for Unstructured Text Data. In *2022 IEEE 4th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA)*. 73–77. https://doi.org/10.1109/ICCCMLA56841.2022.9989047

[117] Sterre van Breukelen, Ann Barcomb, Sebastian Baltes, and Alexander Serebrenik. 2023. "STILL AROUND": Experiences and Survival Strategies of Veteran Women Software Developers. arXiv:2302.03723 [cs.SE]

[118] Sahil Verma and Julia Rubin. 2018. Fairness Definitions Explained. In *Proceedings of the International Workshop on Software Fairness* (Gothenburg, Sweden) *(FairWare '18)*. Association for Computing Machinery, New York, NY, USA, 1–7. https://doi.org/10.1145/3194770.3194776

[119] Zhiyuan Wan, Xin Xia, David Lo, and Gail C Murphy. 2019. How does machine learning change software development practices? *IEEE Transactions on Software Engineering* 47, 9 (2019), 1857–1871.

[120] Jiangtao Wang, Bin Guo, and Liming Chen. 2022. Human-in-the-loop Machine Learning: A Macro-Micro Perspective. arXiv:2202.10564 [cs.HC]

[121] Song Wang, Devin Chollak, Dana Movshovitz-Attias, and Lin Tan. 2016. Bugram: bug detection with n-gram language models. In *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*. 708–719.

[122] Silv Wang, Kai Fan, Kuan Zhang, Hui Li, and Yintang Yang. 2022. Data complexity-based batch sanitization method against poison in distributed learning. *Digital Communications and Networks* (2022). https://doi.org/10.1016/j.dcan.2022.12.001

[123] Steven Euijong Whang, Yuji Roh, Hwanjun Song, and Jae-Gil Lee. 2022. Data Collection and Quality Challenges in Deep Learning: A Data-Centric AI Perspective. arXiv:2112.06409 [cs.LG]

[124] Batia Mishan Wiesenfeld, Yin Aphinyanaphongs, and Oded Nov. 2022. AI model transferability in healthcare: a sociotechnical perspective. *Nature Machine Intelligence* 4, 10 (2022), 807–809.

[125] Xingjiao Wu, Luwei Xiao, Yixuan Sun, Junhang Zhang, Tianlong Ma, and Liang He. 2022. A survey of human-in-the-loop for machine learning. *Future Generation Computer Systems* 135 (oct 2022), 364–381. https://doi.org/10.1016/j.future.2022.05.014

[126] Mingke Yang, Yuming Zhou, Bixin Li, and Yutian Tang. 2023. On Code Reuse from StackOverflow: An Exploratory Study on Jupyter Notebook. *arXiv preprint arXiv:2302.11732* (2023).

[127] Zhou Yang, Muhammad Hilmi Asyrofi, and David Lo. 2021. BiasRV: Uncovering Biased Sentiment Predictions at Runtime. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Athens, Greece) *(ESEC/FSE 2021)*. Association for Computing Machinery, New York, NY, USA, 1540–1544. https://doi.org/10.1145/3468264.3473117

[128] Zhou Yang, Jieke Shi, Muhammad Hilmi Asyrofi, and David Lo. 2022. Revisiting Neuron Coverage Metrics and Quality of Deep Neural Networks. In *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE Computer Society, Los Alamitos, CA, USA, 408–419.

[129] Zhou Yang, Chenyu Wang, Jieke Shi, Thong Hoang, Pavneet Kochhar, Qinghua Lu, Zhenchang Xing, and David Lo. 2023. What Do Users Ask in Open-Source AI Repositories? An Empirical Study of GitHub Issues. In *Proceedings of the 20th International Conference on Mining Software Repositories (MSR '23)*. 12 pages.

[130] Zuhao Yang, Fangneng Zhan, Kunhao Liu, Muyu Xu, and Shijian Lu. 2023. AI-Generated Images as Data Source: The Dawn of Synthetic Era. arXiv:2310.01830 [cs.CV]

[131] Fahri Anıl Yerlikaya and Şerif Bahtiyar. 2022. Data poisoning attacks against machine learning algorithms. *Expert Systems with Applications* 208 (2022), 118101.

[132] Xue Ying. 2019. An Overview of Overfitting and its Solutions. *Journal of Physics: Conference Series* 1168 (02 2019), 022022. https://doi.org/10.1088/1742-6596/1168/2/022022

[133] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A. Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. 2022. Differentially Private Fine-tuning of Language Models. arXiv:2110.06500 [cs.LG]

[134] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. 2021. Large Scale Private Learning via Low-rank Reparametrization. arXiv:2106.09352 [cs.LG]

[135] Beiqi Zhang, Tianyang Liu, Peng Liang, Chong Wang, Mojtaba Shahin, and Jiaxin Yu. 2022. Architecture Decisions in AI-based Systems Development: An Empirical Study. arXiv:2212.13866 [cs.SE]

[136] Jie Zhang, Earl T Barr, Benjamin Guedj, Mark Harman, and John Shawe-Taylor. 2019. Perturbed model validation: A new framework to validate model relevance. (2019).

[137] Jie M. Zhang, Mark Harman, Lei Ma, and Yang Liu. 2019. Machine Learning Testing: Survey, Landscapes and Horizons. arXiv:1906.10742 [cs.LG]

[138] Yuheng Zhang, Ruoxi Jia, Hengzhi Pei, Wenxiao Wang, Bo Li, and Dawn Song. 2020. The secret revealer: Generative model-inversion attacks against deep neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 253–261.

[139] Hui Zhao, Zhihui Li, Hansheng Wei, Jianqi Shi, and Yanhong Huang. 2019. SeqFuzzer: An industrial protocol fuzzing framework from a deep learning perspective. In *2019 12th IEEE Conference on software testing, validation and verification (ICST)*. IEEE, 59–67.