Software Fairness: An Analysis and Survey

EZEKIEL SOREMEKUN, Singapore University of Technology and Design, Singapore
MIKE PAPADAKIS, MAXIME CORDY, and YVES LE TRAON, SnT, University of Luxembourg, Luxembourg

ABSTRACT In the last decade, researchers have studied fairness as a software property. In particular, how to engineer fair software systems. This includes specifying, designing, and validating fairness properties. However, the landscape of works addressing bias as a software engineering concern is unclear, i.e., techniques and studies that analyze the fairness properties of learning-based software. In this work, we provide a clear view of the state-of-the-art in software fairness analysis. To this end, we collect, categorize and conduct in-depth analysis of 164 publications investigating the fairness of learning-based software systems. Specifically, we study the evaluated fairness measure, the studied tasks, the type of fairness analysis, the main idea of the proposed approaches and the access level (e.g., black, white or grey box). Our findings include the following: (1) Fairness concerns (such as fairness specification and requirements engineering) are under-studied; (2) Fairness measures such as conditional, sequential and intersectional fairness are under-explored; (3) Semi-structured datasets (e.g., audio, image, code and text) are barely studied for fairness analysis in the SE community; and (4) Software fairness analysis techniques hardly employ white-box, in-processing machine learning (ML) analysis methods. In summary, we observed several open challenges including the need to study intersectional/sequential bias, policy-based bias handling and human-in-the-loop, socio-technical bias mitigation.

Additional Key Words and Phrases: Software Fairness, Software Analysis, Bias, Discrimination, Artificial intelligence, Machine learning

ACM Reference Format:
Ezekiel Soremekun, Mike Papadakis, Maxime Cordy, and Yves Le Traon. 2025. Software Fairness: An Analysis and Survey. 1, 1 (August 2025), 46 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Software fairness is a property of learning-based systems which aims to ensure that the software does not exhibit biases [204]. Given a set of inputs, a fair software should not result in discriminatory outputs or behaviors for inputs relating to certain groups or individuals. In essence, the goal is to ensure that software systems exhibit fair behavior for all inputs that are similar for the task-at-hand. For instance, *discriminatory inputs*, inputs that are similar for a task but only differ in *sensitive attributes* (e.g., gender, race or age), should produce similar outputs or induce similar program behaviors [81].

Specifically, the goal of *software fairness analysis* is to ensure a given system produces the same results or exhibit similar behaviors for a number of *discriminatory inputs*. As an example, consider a sentiment analyzer software that determines the emotional state or situation in a text, which outputs either a positive emotion (e.g., text describing excitement) or negative emotion (e.g., text portraying sadness or anger). Figure 1 shows an example of a bias in such

Authors' addresses: Ezekiel Soremekun, ezekiel_soremekun@sutd.edu.sg, Singapore University of Technology and Design, 8 Somapah Rd, Singapore, Singapore, Singapore, 487372; Mike Papadakis, michail.papadakis@uni.lu; Maxime Cordy, maxime.cordy@uni.lu; Yves Le Traon, Yves.LeTraon@uni.lu, SnT, University of Luxembourg, 6, rue Richard Coudenhove-Kalergi, Luxembourg, Luxembourg, Luxembourg, L-1359.

Manuscript submitted to ACM

a system, where the output (sentiment) is different when the gender of the noun in the text is a "man" compared to a "woman". This is an illustrative example of (gender) bias found in real-world natural language processing (NLP) systems [17, 143, 190].

Several researchers have studied software fairness analysis, with the aim to address a fundamental question: How to engineer fair software systems? Such works consider fairness as a non-functional software property and a software engineering (SE) concern. This includes work studying how to specify [21, 185], test [81], and mitigate [44] fairness properties in software systems. However, despite several works on software fairness analysis, it is difficult to understand the state of research practice: Specifically, what fairness concerns have been addressed? How have they been addressed? What are the open problems and challenges?

In this paper, we aim to provide a clear view of the state-of-the-art in software fairness analysis, i.e., techniques and studies that analyse the fairness properties of learning-based software. This paper aims to analyze the trends in software fairness analysis, the available techniques, the focus of the research community, the problems that have been addressed and the open research problems. To this end, we perform a systematic analysis of the literature, where we studied 164 papers studying *fairness as a software property*. These papers are mostly published in fairness-related or software engineering (SE) venues, as well as venues focused on machine learning (ML), artificial intelligence (AI), security, computer vision (CV) and natural language processing (NLP). Particularly, we conduct an in-depth study of the set of publications that explore fairness with the lens of software engineering, e.g., in terms of software quality control, requirement engineering, design and development. We then characterize several factors encapsulated in these research papers, including the evaluated fairness measure (e.g., individual, group, causal or conditional fairness), the studied tasks (e.g., credit rating, CV, NLP), the type of fairness analysis (e.g., testing or mitigation), the main idea of the proposed approach and the level of access (e.g., black, white or grey box).

Overall, we observe that the research community is facing several open challenges in addressing the following fairness concerns: compounding or intersectional bias, verification of fairness properties, sequential bias, equity-based bias handling and socio-technical solutions to bias. The key findings of this work includes the following:

- Software Fairness analysis is mostly performed to validate or mitigate biases, i.e., the focus of the community has been to test and reduce unfair program behaviors, other fairness concerns (such as requirements engineering and verification) are under-studied;
- The most studied fairness measure include individual, group and causal fairness, measures such as conditional, sequential and intersectional fairness remain under-explored;
- The most examined tasks involve tabular datasets (such as the adult income dataset), while semi-structured datasets (e.g., audio, image and text) are barely studied in the SE community;
- The most employed techniques in the SE community are mutation (e.g., input perturbation) and specification (e.g., using input templates, schemas or grammars) -based techniques, other approaches such as in-processing machine learning (ML) analysis methods are hardly employed for software fairness analysis;
- Most approaches support the analysis of an atomic attribute (e.g. race, or gender), very few approaches study the combination or sequence of attributes (e.g., race × gender), the compounding effect of multiple instances of an attribute, or complex attributes (such as non-binary gender);
- We found little or very few works tackling the fairness concerns like fairness test metrics/adequacy, automatic repair of biased classifiers or time-based fairness concerns (e.g., sequential or regression fairness bugs).

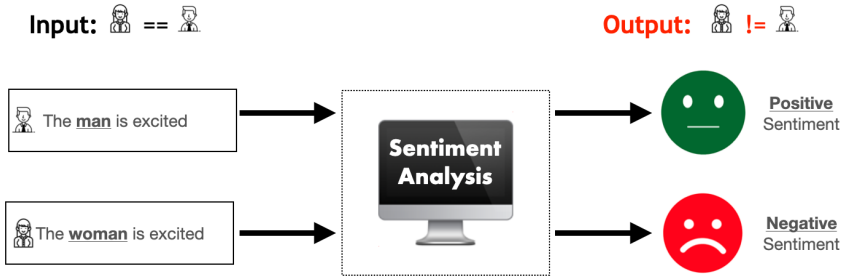


Fig. 1. Example of Individual Fairness Violation using a Sentiment Analysis AI System

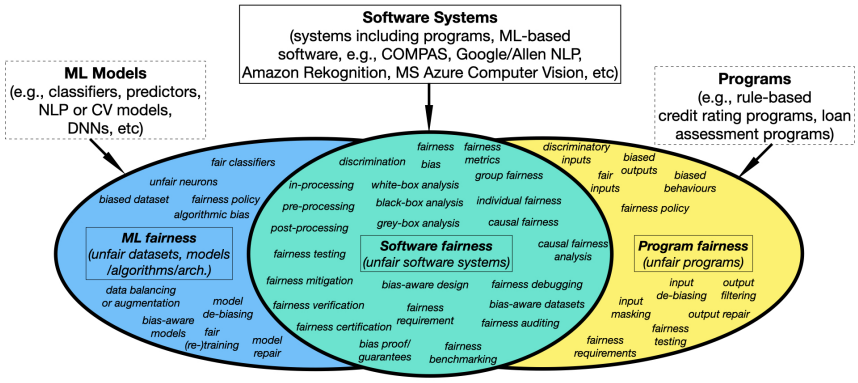


Fig. 2. Taxonomy and interplay of ML fairness and Software Fairness

The rest of this paper is organised as follows: We provide background on software fairness in section 2. We discuss the process of collecting/analyzing publications in section 3, and subsection 4.3 highlights our research questions. In section 5, we provide in-depth analysis of our findings, and section 6 discusses the limitations and threats to validity of our study. Finally, we discuss open research challenges (section 8) in software fairness analysis and conclude (section 9).

2 BACKGROUND

We provide background on software fairness, with the definition of terms employed throughout the paper, illustrative examples and a discussion of closely related works.

2.1 Definition of Terms

We describe the main terms used in the rest of this paper, and the context in which they apply to our analysis and survey.

```java
public static boolean approve_credit(int age,  boolean is_married,  int salary,
int years_of_experience) {
        int salary_threshold =  50000;
        int min_age =  30;
        int max_age =  60;
        int experience_threshold =  5;
        boolean credit_approval = false;

        if (!is_married || years_of_experience < experience_threshold){
            return credit_approval;
        }

        if (salary > salary_threshold && age > min_age && age < max_age){
            credit_approval = true;
            return credit_approval;
        }

        return credit_approval;
}
```

Fig. 3. Illustration of an *unfair* rule-based credit approval program which uses protected attributes ("age", "marital status")

Bias: In this paper, we refer to *bias* in terms of algorithmic bias, specifically, bias occurs when a software system *systematically* and *unfairly* discriminate against certain individuals or groups of individuals in favour of others [78]. Algorithmic bias causes discrimination against certain people and can lead to real-world harms in terms of the representation or allocation of resources to such individuals or groups [53].

Software Fairness: There has been a significant work in understanding and defining *fairness* as a software behavior [204]. In this work, we examine *fairness as a software property*. The aim is to focus on works examining *software fairness via the lens of software engineering*, e.g., how to engineer bias-aware software or prevent bias in software systems. To this end, we study papers that study fairness as a SE concern including in terms of a requirements engineering [24], software quality control [81], software design [116], and software verification [8].

2.2 The interplay of ML Fairness and Software Fairness

In this section, we discuss the interplay (similarities, differences and intersections) of ML fairness and software fairness, e.g., in terms of their *design (components and pipelines)*, *terminology*, and *analysis*. Figure 2 further provides a venn diagram describing the taxonomy and interplay of ML fairness and software fairness.

Program Fairness: This is the fairness property of a computer program. A program may exhibit unfair behaviors or produce unfair outputs. As an example, consider the rule-based credit rating system in Figure 3. Let us assume that the fairness policy of an institution or a country protects "*age*" and "*marital status*", such that they can not be used to determine the credit approval rating of an individual. Then, we say this program is *unfair* because its credit approval is computed using protected attributes – *age* and *marital status*. Figure 4 further shows an example input where the

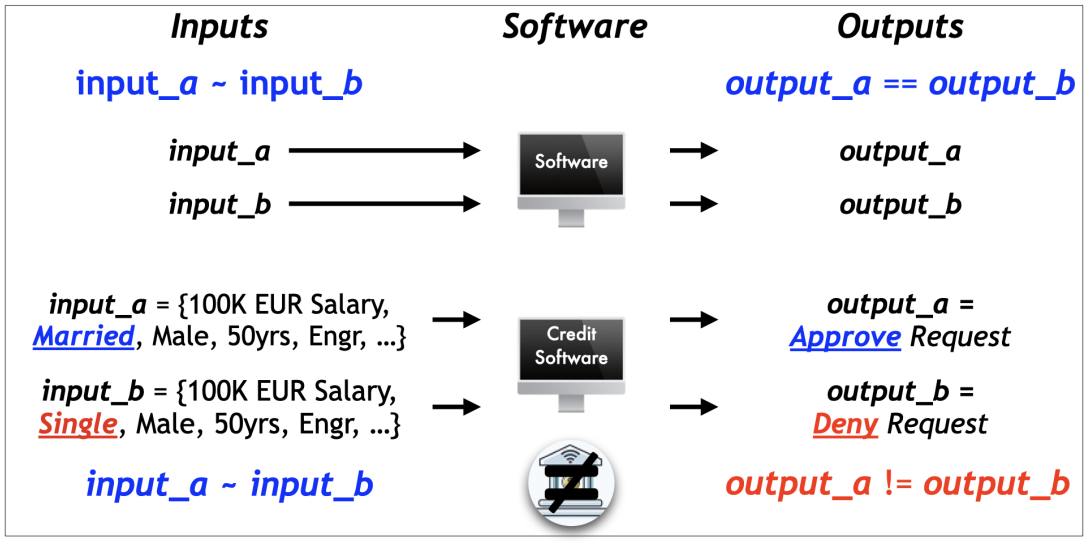


Fig. 4. Program fairness property showing fair program behavior (`output_a == output_b`) and unfair program behavior (`output_a != output_b`) using a (credit approval) software

program violates the defined fairness policy. Particularly, the conditional checks ("`if`" condition statements in lines eight (8) and twelve (12) in Figure 3) violate the aforementioned fairness policy.

ML Fairness: *ML fairness* refers to the fairness property of a *machine learning (ML) model*, e.g., an *unfair ML model* may produce unfair predictions/classifications. For instance, an ML classifier can be considered *unfair* if its prediction for a *protected* group is significantly different from that of *other* groups. Consider the sentiment analyzer in Figure 7, let us assume it is *only* an ML model (with no additional components), then we say that the ML model is unfair since it produces significantly different outcomes for female-biased occupations (G_1) versus male-biased occupations (G_2).

Software Fairness: (Un)fairness may stem from one or more components of a software system. Indeed, *unfair* behaviors may be from the *ML model* itself, other software components, multiple components or the interaction of the ML model with other software components (*see* Figure 5). For instance, consider the sample ML-based software in Figure 5. On one hand, ML fairness in this system may be caused by biases inherent in the ML model, e.g., its training data or algorithms. On the other hand, software *unfairness* may stem from any of the components of the system, including the ML models, and other software components – e.g., input/output validators, or servers. Besides, unfair software behaviors may stem from the interaction of multiple components, e.g., the response processing from the ML model to the response/output servers and validators. For instance, a software whose input validator or tokenizer can only process input up to a limited input size (e.g., due to memory constraint) may induce unfair behaviors for very long inputs. As an example, some NLP systems (e.g., BERT-based models) only process the first N (e.g., 512) input tokens [218], and such input/memory constraints may cause unfair behaviors despite the model's capability or fairness properties.

Similarly, consider the credit approval software in Figure 4. Let us assume that marital status is a protected attribute. Then, we say that the credit approval software is *unfair* since it treats two individuals differently even though they have

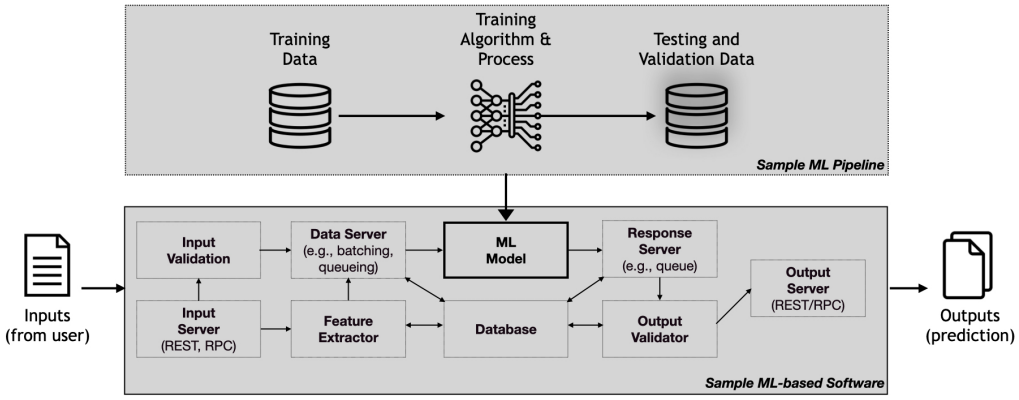


Fig. 5. An illustration of ML-based software systems (adapted from Lewis et al. [129], Muccini and Vaidhyanathan [152])

similar characteristics except for the protected attribute (marital status). In particular, the software approves the credit request of the *married individual* (`input_a`) but denies the credit request of the *single, unmarried individual* (`input_b`). We note that this behavior could be as a result of different sources depending on the implementation of the system, i.e., (a) a program implementation (e.g., rule based system in Figure 3), (b) an ML model (e.g. , automated credit rating classifier , or (c) a combination of multiple components including programs and ML models (e.g., Figure 2). Overall, we consider all aforementioned settings as an unfair software behavior, since it is the behavior of the entire software system regardless of the source. Thus, in our setting, software fairness applies to all of the three aforementioned configurations of the software.

White-box Setting: In a *white-box setting*, where users or developers have access to the software implementation, it is possible to attribute the unfair software behavior to a specific component or analyze a specific sub-component for fairness properties. In this work, if fairness analysis involves examining the internals of a specific sub-component of the software, e.g., *only* the ML model or the rule-based program, then we consider this as *white-box fairness analysis*. This includes the analysis of a specific sub-component, e.g., the ML model (e.g., sentiment analyzer Figure 7) or a specific program (e.g., the program in Figure 3). For instance, white-box fairness analysis includes examining the training process or architecture of an ML model to improve the fairness properties of the model, as well as the static/dynamic analysis or testing of a program to improve its fairness properties.

Black-box Setting: In a *black-box setting* where a customer or the developer has no access or knowledge of the system that is deployed, it is difficult to determine the component(s) responsible for the unfair behavior. Firstly, in a black-box setting, it is often *unknown* if the system is implemented as a learning-based system (e.g., ML model), a program (e.g., the rule based program in Figure 3), or a combination of both. This is because several ML-based software and AI systems are proprietary, black-box systems (e.g., COMPAS system, Google NLP, Amazon Rekognition, etc). They mostly only provide a web or API access and do not provide information on the implementation of the system, or its architecture. Secondly, in a black-box setting, it is difficult to determine the unfair component in a complex system, even when it is known that the system is composed of an ML model and/or a program. When analyzing an ML-based software system (e.g., Figure 5) in the black-box setting, it is difficult to determine if the ML model or rule-based program

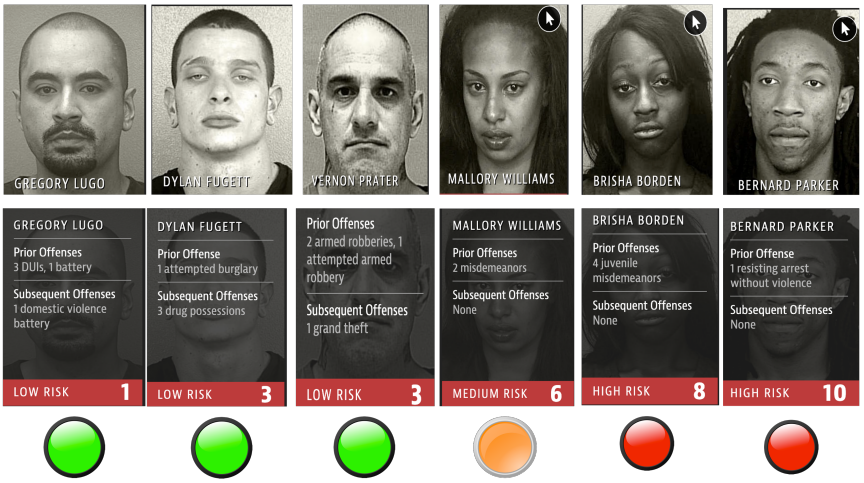


Fig. 6. An illustrative example of real-world discrimination (unfair software behavior) in the COMPAS software, a program that predicts recidivism – the likelihood of committing a future crime (adapted from Angwin et al. [13, 14])

is responsible for the unfair behavior, or whether its interaction with other components caused unfairness or even other components are responsible for the unfair behavior. However, in this work we consider the analysis of such systems as software fairness analysis since, despite the lack of knowledge/access, the system is delivered/deployed as a software. For instance, the popular COMPAS system, which has been shown to be biased towards certain race and gender attributes, is a black-box proprietary software system [13, 14]. Figure 6 illustrates the biases exposed in this system, where *black* and *female* convicts are predicted to have a higher probability of recidivism (risk of re-committing crimes), even when their previous crimes are less serious than their *white* and *male* counterparts [13, 14]. We consider the analysis that reveals biases in COMPAS as a *black-box fairness analysis*, since the researchers had no access to the training process, implementation or software architecture of the system [13, 14].

Grey-box settings: In this work, grey-box techniques refers to methods and tools that employ a combination of the aforementioned white-box and black-box methods for fairness analysis. Table 3 details such grey-box approaches. For instance, Tizpaz-Niari et al. [201] proposed a technique that leverage both the input space and the model analysis for fairness testing.

White-box versus Black-box in ML Model-only settings: We note that when testing only the ML model, researchers often refer to white-box or black-box analysis as approaches that either inspect the ML model or not, respectively. In such settings, this work (Table 3) considers inspecting model internals (e.g., neurons) as white-box analysis, especially when only the model is tested. Conversely, black-box analysis refers to fairness analysis without inspecting the model internals, in ML model-only settings. Subsequently, approaches that combine both approaches (i.e., leveraging analysis/information from both within and outside the ML model) are referred to as grey-box approaches.

2.3 Fairness Metrics

Verma and Rubin [204] provides comprehensive definitions and categories of several fairness metrics employed in the literature. Figure 1 and Figure 7 exemplify the two most popular metrics, namely *individual fairness* and *group fairness*. In the following, we define these two metrics.

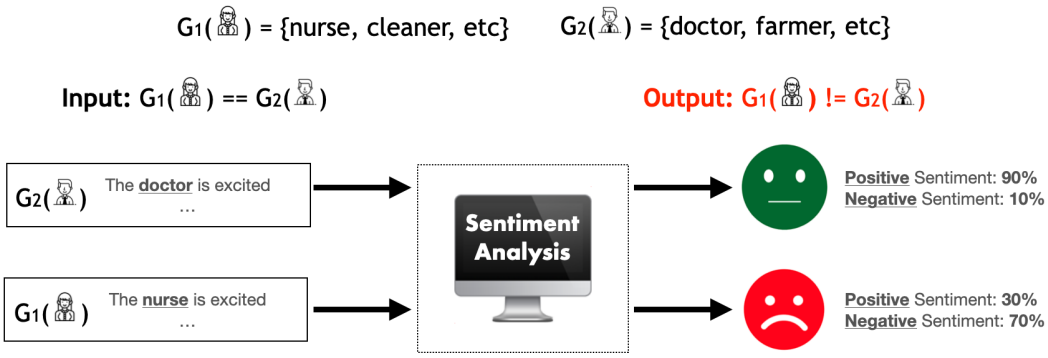


Fig. 7. Example of Group Fairness Violation using a Sentiment Analysis AI System

Individual fairness: Dwork et al. [62] provides a comprehensive definition of individual fairness. Individual fairness means that the software should treat similar individuals similarly. Consider the sentiment analysis system in Figure 1, it violates individual fairness since it treats a text with a "man" as a *noun* differently from that with a "woman". This metric requires that the individuals should be similar for the purposes of the respective task and the outcomes should have similar distributions. Formally, individual fairness is a violation of the following condition:

$$\left|f(a) - f(a')\right| \leq \tau \tag{1}$$

Here, a and a' are similar individuals (inputs), f is a software esystem (e.g., automated classifier) and τ is some threshold which is chosen using the inputs and the model as context. In our example (Figure 1), the text containing "man" or "woman" should be treated similarly as individuals, however the output shows that the system provides different outcomes for the similar inputs.

Group fairness: A system satisfies group fairness if subjects in the protected and unprotected groups have equal probability of being assigned a particular outcome [204]. This fairness metric aims to ensure that two or more groups are treated similarly. Consider the sentiment analysis system in Figure 7, it violates group fairness since it treats texts containing nouns belonging to *male-biased occupations* (e.g., doctor, farmers etc) differently from similar texts that contain *female-biased occupations* (e.g., nurse, cleaner, etc). Formally, group fairness is maintained if the following condition is true:

$$Pr(f(a) = +|A = a) = Pr(f(b) = +|A = b) \ \forall a, b \in A \tag{2}$$

Given equivalent inputs from different groups a and b, the aforementioned definition checks for the equivalence of the outputs from software f. Here, the choice of a group is determined by random variable A and the positive prediction rate is denoted by $+$. In this example, the groups are *male-biased occupations* and *female-biased occupations*, and the model produces significantly different distribution of positive and negative predictions for each group. As an example, a group fairness violation is that texts containing male-biased jobs are more (90%) likely to return a positive sentiment than female-biased jobs (30%).

2.4 Related Work

In the last few years, several researchers have surveyed the problem of bias or fairness in learning-based software, but these surveys have been mostly *specialised*, i.e., they studied this problem for a specific domain, bias, or metric. Concretely,

previous surveys on fairness analysis target a particular sub-domain (e.g., NLP [32, 84], CV [64], ranking [161, 223] or finance [171]), a specific sensitive attribute or bias (e.g., race [74]), or a specific fairness metric (e.g., causal fairness [147] or intersectional fairness [89]). Thus, these surveys do not account for the advances in other domains, attributes and metrics.

Other fairness surveys are either more general beyond fairness concerns or are focused on specific goals or goals beyond SE. For instance, some surveys explore general testing of ML systems for several properties [227], other surveys providing a taxonomy or review of bias in ML systems [30, 148, 165, 197] and some explore how to handle bias in special circumstances, e.g., in the absence of demographic information [15]. More importantly, none of these papers have performed a systematic analysis of bias in the context of *software engineering* or *the analysis of software fairness*. Specifically, studies and methods that *(automatically) evaluate software fairness properties in learning-based systems*. Our survey focuses on the literature w.r.t. to several aspects of SE, thus involving a significant number of SE-related publications. In the following we discuss the closely related work to this paper, in particular the published surveys in this area.

There are a some *domain-specific* surveys of fairness where researchers have surveyed fairness issues in a specific sub-domain of learning-based software, e.g., recommendation systems [63, 93–95, 167], ranking-based systems [223, 224], sequential decision systems [233], NLP [32], CV [64], or financial services [171]. Blodgett et al. [32] comprehensively analyzed 146 papers addressing fairness of NLP systems, especially quantitative techniques for measuring or mitigating bias. The authors found that almost all surveyed papers are poorly matched to their motivations and do not engage with the relevant literature outside of NLP. The authors recommend that fairness analysis in NLP systems should be based on characterizing system behaviors that are harmful, by centering bias mitigation around people and their experiences. Fabbrizzi et al. [64] provides a survey on bias in visual datasets, i.e., for CV applications. The authors describe different biases in visual datasets, the proposed methods for detecting and measuring these biases and existing bias-aware datasets. The authors concluded that there is no bias-free dataset and detecting biases in visual datasets is an open problem and recommend a checklist to help practitioners spot different biases in their dataset, in order to make bias explicit. Ashurst and Weller [16] surveyed approaches for fairness without demographic data, discussing the benefits and limitations of the approaches. Fabris et al. [67][66] also surveyed the datasets used in algorithmic fairness research. In addition, the authors present a search engine to search amongst the surveyed datasets for algorithmic fairness [65][1]. Similar to this paper, the authors found that Adult, COMPAS, and German Credit are the three most popular fairness dataset. The paper further highlights the weaknesses of these datasets and propose alternative datasets for fairness practitioners and researchers. Meanwhile, Zehlike et al. [223] studied the application of fairness-enhancing interventions in ranking algorithms, especially examining the literature on incorporating fairness requirements into algorithmic rankers. The authors surveyed papers from several venues, including data management, algorithms, information retrieval, and recommendation systems. The goal of the survey is to provide a new framework that unifies fairness mitigation objectives and ranking requirements, such that it allows to examine the trade-off between both goals. Ekstrand et al. [63] studied how algorithmic fairness issues applies to information retrieval and recommendation systems, especially addressing how to translate algorithmic fairness from classification, scoring, and ranking settings into recommendation and information retrieval settings.[2] In addition, Zhang and Liu [233] provides a literature review of fairness in sequential learning-based decision-making systems, i.e., systems where decision-making are not a one time event, but rather occur in a sequential nature, such that decisions made in the past may have an impact on future data. Unlike these papers, this

[1] http://fairnessdata.dei.unipd.it/
[2] https://fair-ia.ekstrandom.net/

work is not *domain-specific*, instead, we study the problem of fairness analysis across several (sub-)domains, including all of the aforementioned domains.

Moreover, other surveys on fairness are *bias-specific* or *metric-specific*, they either focus on a specific sensitive attribute (e.g., race) [74], or a specific fairness metric (e.g., causal fairness) [147], respectively. For instance, Field et al. [74] provides a survey of race-related bias in the stages of NLP model development. The authors surveyed 79 papers with the goal of understanding the gaps between *race-related bias* analysis in NLP and other related fields. The authors found that race has been siloed as a niche topic in the NLP community and often ignored in many NLP tasks. The authors also emphasize the need for racial inclusion in NLP research. In addition, Makhlouf et al. [147] study the application of causality to address the problem of fairness, by studying papers that examine *causal fairness* properties and their applicability in real-world scenarios. The authors employed *identifiability theory* to determine the criteria for the real-world applicability of *causal fairness*. However, in this work, we examine papers examining several sensitive attributes, biases and fairness metrics. In particular, unlike these metric or bias -specific surveys, we do not focus on papers examining a single type of bias or fairness metric. Instead, we evaluate the literature across different biases and fairness metrics, including race and causal fairness, respectively.

A few researchers have conducted surveys of the literature on software fairness, albeit mostly targeting fair prediction in machine learning [41, 80, 148, 155], general ML testing [227] or fairness notions across domains [108] or in specific domains, e.g., concurrent systems [127]. For instance, Gajane and Pechenizkiy [80] studied the *formalization* of fairness in the machine learning literature for prediction tasks, especially examining how this relates to the notions of distributive justice in the social science literature. In their study, the authors proposed that two notions of distributive justice be formalised for fairness in ML, namely *equality of resources* and *equality of capability of functioning*. Likewise, Ntoutsi et al. [155] provides an introductory survey on the technical challenges and available solutions to bias in data-driven AI, with a focus on the legal grounds for bias challenges and the societal implications of these solutions. Boykin et al. [34] surveyed the intersection of ML and psychology, specifically with a focus on psychological mechanisms underlying fairness preferences. Cheng et al. [50] conducted a systematic review of socially responsible AI algorithms with the aim of examining the literature beyond algorithmic fairness and connecting major aspects of AI to societal well-being.

Mehrabi et al. [148] examined the fairness issues in real-world applications, specifically investigating the different sources of bias in AI systems and providing a taxonomy of fairness definitions in the ML community. Similarly, Caton and Haas [41] provide an overview of fairness mitigation approaches for ML, by categorising mitigation techniques into several stages in the ML model development pipeline. The authors highlight 11 mitigation methods categorised into three areas, namely pre-processing, in-processing, and post-processing methods. In addition, Zhang et al. [227] and Chen et al. [46] surveyed the testing of several properties in machine learning (ML), including fairness properties. In contrast to these works, we focus on fairness properties as it relates to the SE development pipeline. Beyond fairness mitigation and testing, we examine works studying fairness formalization, empirical evaluation, detection and improvement.

Some researchers, Hutchinson and Mitchell [108] and Kwiatkowska [127], have also studied the notions of fairness properties across domains, and for concurrent systems, respectively. Notably, Hutchinson and Mitchell [108] surveyed the history of the definitions of fairness properties over the last 50 years across multiple disciplines, including education, hiring, and machine learning. The authors compared past and current notions of fairness along several dimensions, including the fairness criteria, the purpose of the criteria (e.g., testing) and how it relates to the mathematical method for measuring fairness (e.g., classification, regression) and people (individuals, groups, and subgroups). Unlike the aforementioned works, our survey of fairness is *more general*, we study software fairness beyond advances in specialised ML communities. In this work, we additionally examine several papers from security, CHI, PL, CV and NLP venues.

Overall, we provide a systematic literature review of the analysis of software fairness properties for learning-based systems. To the best of our knowledge, this work is the only systematic literature review concerned with the *analysis of fairness a (non-functional) software property of learning-based systems*. We are not aware of any other survey that focuses on this research area, i.e., software fairness analysis, especially providing a comprehensive survey of its formalization, testing, diagnosis and mitigation across several fairness metrics, biases and domains.

3 METHODOLOGY

The research methodology employed in this work is based on the methodology detailed in Kitchenham [125]. In the following, we provide the details of our research protocol:

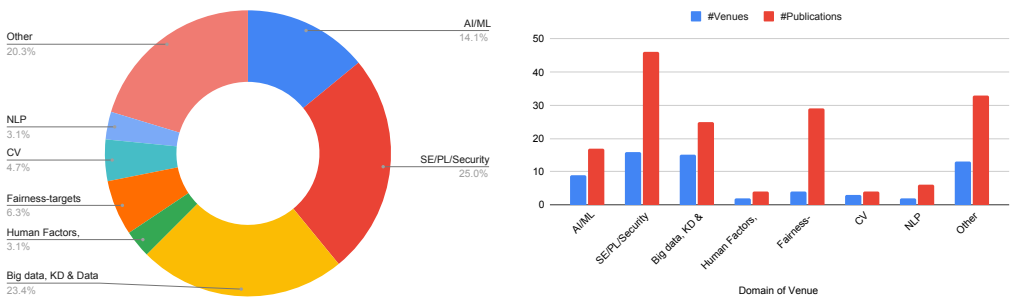
(1) **Aim and Scope:** First, we define the goals of this work and the scope of works to be examined. Then, we define the scientific questions, the analysis protocol and the information relevant for our data analysis. To this end, we define the research questions (*see* subsection 4.3).

(2) **Detailed Information:** To analyse each paper in-depth, we identify the information necessary to answer all research questions, this includes information that allow us to categorise, understand and describe the problem addressed by each approach, and the technique or results provided by each paper. This informed the publication search (e.g., our focus venues,), as well as the keywords employed in the filtering process used in identifying relevant papers. Among several details, our interests include the studied fairness metrics and biases, the form of fairness analysis (e.g., fairness testing) and the level of access (e.g., white, grey or black -box).

(3) **Publication Search:** We curated fairness-related publication via three means, (1) we searched the top venues in SE (such as ICSE, TSE and FSE), Programming Languages (PL) (e.g., PLDI and POPL), Security (e.g., CCS and Euro S & P), Artificial Intelligence (AI) (e.g., AAAI), ML (e.g., ICML, NeurIPS), CV (e.g., CVPR) and NLP (e.g., ACL, EMNLP), (2) we collected papers from fairness focused conferences and workshops such as FairWare, FAT, FATML, and FaaCT; and (3) we conducted a keyword guided publication search of paper respositories (such as ACM Digital library[3], IEEE Xplore Digital Library[4] and Google Scholar[5]) using popular fairness-related terms such as “bias”, “fairness”, “discrimination”, etc. In total we merged all collected papers which amounted to 420 papers from 65 venues. In addition, we conducted a focused search of more recent works from SE venues since our initial study. We collected an additional 72 publications mostly from the top SE venues from 2022-2023.

(4) **Filtering:** To identify relevant publications we filter out publications that are not relevant to our goals and analysis. Specifically, we filter out papers that (1) do not conduct fairness analysis as a part of the software engineering process, i.e., papers that are not relevant to the requirements, design and quality control of AI or ML -based software systems; (2) we exclude papers that are not written in English language, duplicate papers, as well as short papers or extended abstracts; (3) we also exclude papers that analyze fairness for other systems, i.e., fairness for non-learning-based software systems. For instance, we exclude works on fairness of non-software-based systems (e.g, transport systems [174], food inspection systems [188]); (4) we also exclude papers that are not focused on software fairness properties, e.g., papers studying other properties such as robustness, consistency, security or accuracy; (5) finally, we read the abstract of each paper and excluded papers that are not research papers studying software fairness, for instance, we excluded surveys, literature reviews and invited lectures. From our initial search, we filtered out 256 papers and were left with 164 papers for our initial analysis. In our

[3] https://dl.acm.org/
[4] https://ieeexplore.ieee.org/Xplore/home.jsp
[5] https://scholar.google.com/



(a) Publication Distribution per domain

(b) Number of Publications and Venues per Domain

Fig. 8. Details of Publication domains

follow-up search of the recent SE literature (2022-23), we filtered 31 papers from the 72 collected publications. This resulted in the analysis of 41 additional papers.

In the course of both publication searches, i.e., the initial search (2010-2021) and SE-focused search (2022-2023), we found a total of 492 papers (420 in first round and 72 in the second round), and filtered out a total of 287 publications (256 in first round and 31 in the second round). This resulted in a total of 205 analysed papers, i.e., 164 papers in the first round and 41 papers in the second round.

Given the large number of collected papers, we designed a research protocol to analyse each paper and extract certain information from the papers. For each paper, we extract both *the metadata* of the paper, as well as the *detailed research information*. In terms of metadata, we extracted the author details, the paper title, the year and venue of publication and the domain of the publication (e.g., SE, security or PL). For detailed research-relevant information, we studied the following details about the techniques and evaluation of the paper: the employed datasets, the studied fairness metrics, the studied biases (i.e., sensitive attributes, e.g., race), the form of analysis (e.g., fairness testing), the access level, the specific problem addressed, the main idea of the paper, the proposed solution, the resulting findings, as well as the strengths and weaknesses of proposed analysis/solution.

Categorization and Coding Protocol: In our analysis, one researcher collects, analyses and filters the publications following the aforementioned research protocol. For each paper, the researcher collects the metadata and documents all collected papers and their analysis then forwards all details to one other researcher for validation and inspection. For internal validation, at least one other researcher inspects the categorisation of the papers, takes note of conflicts or missing information and informs the researcher. Next, conflicts are resolved by organising a meeting to discuss the differences in the categories, naming of categories, collected data and ascribed descriptions. Finally, to ensure publications are correctly classified, we externally validate our findings and descriptions by sending a draft of our paper to all (cited) authors to provide feedback on mis-catergorization or wrong characterization of their work.

4 EVALUATION SETUP

4.1 Data Collection and Analysis

Our initial analysis involved 164 publications (from 2010 till 2021). Most of the research findings in section 5 (RQ1 to RQ5) are based on the analysis of the initial set of papers. In our follow-up analysis, we examined an additional 41 publications resulting from filtering out 31 papers out of 72 collected papers. These publications were collected from a

more focused search of the recent SE literature (2022 till 2023.) We examine these works and shed light on the progress made since our initial analysis. We further discuss these newly published works and mention the domains of these new works and how they relate to our initial findings in our result discussions and tables.

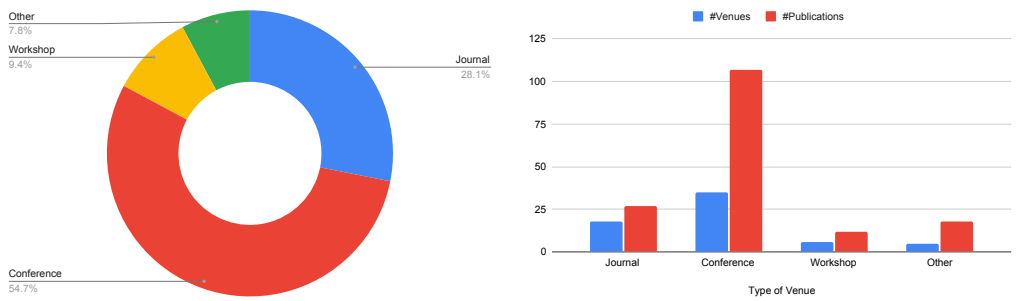
In the initial collection, we examine publications (164) that *analyse software fairness in learning-based systems*. Table 1 provides details of some of the collected publications. We analyse publications from different domains and venues, including SE, PL, AI/ML, CV and NLP (*see* Figure 8). We then perform an *additional focused paper collection* from 2022 to 2023 that is mostly focused on the recent works published in the top SE venues including 2022 and 2023 proceedings of ICSE, ESEC/FSE, TOSEM ISSTA and EMSE. The goal of this collection is to examine the recent research progress since our initial analysis.

We analyze the volume of publications. We first examine the metadata of our publication corpus to determine the volume of publications in different domains and venues. For this analysis, we collected a corpus of 164 papers from 64 different venues and eight (8) different academic domains/fields. Table 1 and Figure 13 show the details of publication venues and the distribution of the publications in our corpus by venue. We are also interested in analysing the details of the publications in terms of the type of venues (e.g., conference, journals and workshops) and the domain (or field) of publication venues (e.g., SE or AI/ML). Figure 8 and Figure 9 show the details of our publications in terms of domains and venue type, respectively. In section 5 and section 4, we further discuss recent publications in constrast to our initial study (these are also highlighted in brackets in Table 2 and Table 7).

4.2 Initial Publication Details

Volume and Domain of Publications: Our analysis showed that even though algorithmic fairness is studied across different research communities, *the study of fairness as a software property is mostly dominant in the SE and data-centric venues*. Notably, *about half (about 48.3%) of all publications on software fairness are in the SE (e.g., FSE) and data-centric (i.e., Big data, knowledge discovery and data mining) venues (e.g., KDD).* Followed by typical top-tier AI/ML domains (e.g., AAAI and NeurIPS) which account for about 14.1% of all papers on software fairness. Figure 8(a) further illustrates that the top software engineering venues (e.g., ICSE, FSE, ISSTA, ASE, TSE, OOPSLA) account for about one in four (about 25.0% of) publications, with this community accounting for most of the venues and publications (*see* Figure 8(b)). In addition, we observed that most papers in our corpus are published in three main venues, namely ICSE (6.1%), FSE (7.9%) or TSE (9.8%). Other popular venues include Big Data, Knowledge Discovery and Data Mining domains (23.4%), as well as AI/ML domains (14.1%) (*see* Figure 8). Likewise, fairness-focused venues (such as FaccT, FATE and AAAI AIES) account for about 6.3% of venues and about 17.7% of all publications. Meanwhile, more specialised venues (e.g., ECCV and CAV) had the least number of works on software fairness. This shows that even though research works on software fairness analysis are published across different domains, most publications are in SE venues. This demonstrates the growing interest in software fairness within the SE community. In particular, the need to apply SE processes, methods and tools to study the implications of fairness measures in practice has become paramount in the SE community. In summary, almost half (48%) of software fairness analysis works are published in typical SE venues and Data-centric venues, this is followed by top-tier AI/ML venues which account for about 14.1% of all publications.

Publication Trends over time: Examining the volume of publications over the years, we observed that *the number of publications in software fairness analysis has been steadily increasing over the last decade*. Figure 13(b) highlights the trend over the last 13 years, it shows that the number of publications in software analysis has been steadily increasing over time, particularly in the last half decade. This trend signifies the growing research interests in fairness analysis in the SE research community. Particularly, a major surge in publications can be observed starting from 2017 till 2021.



(a) Distribution of works by venue type

(b) Number of Publications and Venues per Publication type

Fig. 9. Details of the type of Publication Venue (e.g., Conference or Journal)

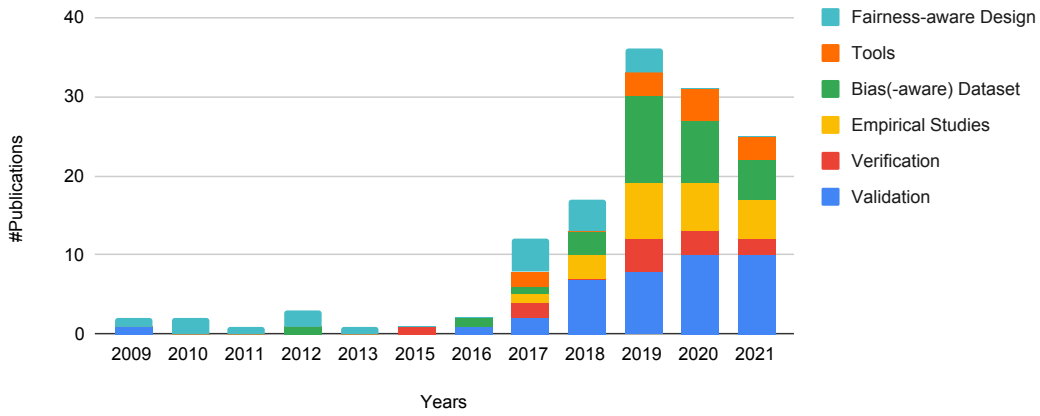
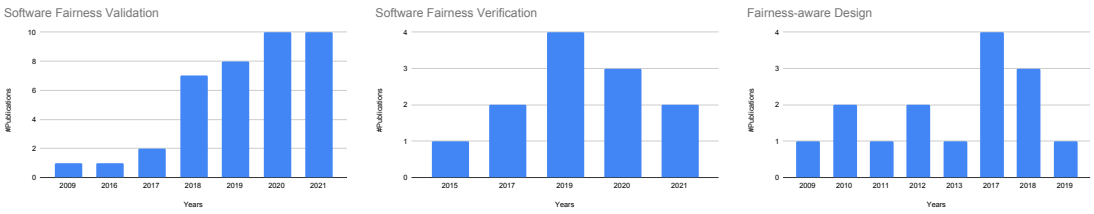


Fig. 10. Detailed Publication Trend by Year

This is following the publication of Galhotra et al. [81] which *first formalizes causal fairness as a non-functional property of (learning-based) software systems.* Indeed, almost all software fairness papers analysied in this paper (152 papers, 92.68%) were published starting from 2017. This demonstrates the growing interests and increasing number of research output in this area over the years.

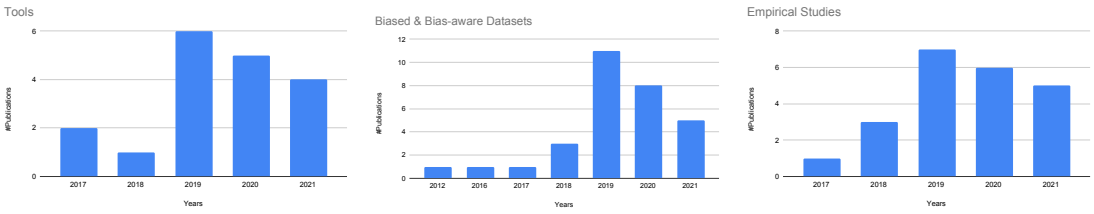
Type of Publication Venues: Figure 9 illustrates the distribution of the type of publication venues in our corpus. We observed that the majority (54.7%) of the publications on fairness analysis are in conference proceedings (*see* Figure 9 (a)). Figure 9(b) also shows that conferences are the most popular venues for software fairness publications, especially top-tier venues like ICSE and FSE, as well as popular conferences focused on Fairness (e.g., FaccT). Journal publications account for about 28.1% of all publications, with TSE, EMSE and JSS being the most popular journal venues. The most popular workshop venue for fairness analysis is FairWare. These findings show that software fairness has (more recently) become an important research area for top-tier SE conferences and journals.

Advances: We analyse the advances made in the analysis of software fairness by focusing on six major areas of fairness concern and their advances over time. Specifically, we analyse the trends in engineering concerns in software fairness



(a) Trends in Fairness Validation (b) Trends in Fairness Verification (c) Trends in Fairness-aware design

Fig. 11. Details of trends in Fairness Verification, Validation and Design



(a) Trends in Fairness Tools (b) Trends in Bias-related Datasets (c) Trends in Empirical Studies

Fig. 12. Details of trends in Fairness Tooling, Datasets and Empirical Studies

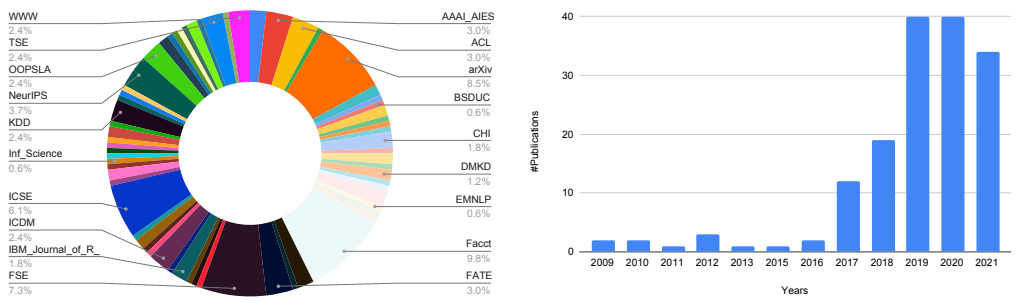
such as validation, verification, design, empirical studies, tooling and datasets. Figure 10 provides a detailed overview of the trend and advances in publications for all of these six concerns. Figure 11 and Figure 12 also provide detailed distribution of publications in each area over the years.

Generally, the number of publications has increased over time for all six SE areas. However, there has been major increase in publications in the area of fairness validation (*see* Figure 11(a)), bias-aware datasets and empirical studies (*see* (*see* Figure 12 (b) and (c)). This is evident by the number of publications (about eight to 12 publications yearly) in these areas in the last half decade (since 2018). In addition, analysing the span of publications also show that fairness-related validation and datasets have been studied for about seven consecutive years, while empirical studies in fairness have only recently become prominent (i.e., in the last 5 years).

Meanwhile, other SE concerns such as the verification, tooling and design of fair software systems have received little attention. These areas have seen fewer (four to six maximum) publications in the last years. However, the span of publications in the design of fair software is larger, with at least a 10 year spread (*see* Figure 11(c)), showing that there has been a steady interest in this area over the years. This implies the need to investigate and examine approaches and methods to address the under-studied areas, especially the verification and tooling of fairness-aware software systems. There has been advances in several areas of software fairness over the years, but most consecutive publications has been in fair learning and validation, with up to 10 yearly publications in the last five years.

4.3 Research Questions

Firstly, we investigate the purpose of these publications including the main idea of the proposed techniques as well as how researchers study software fairness as a software engineering task (RQ1). For instance, we examine if the aim of the proposed method or analysis is to formalize, test, mitigate or diagnose fairness issues in learning-based



(a) Number of Publications per venue

(b) Distribution of Publications over time

Fig. 13. Distribution of Publication Venues and Year of Publications. (Table 1 provides an overview of venues, note that some sectors are unlabeled in Figure 13(a) due to space constraints.)

Table 1. Excerpt of Publication Details ("#" means "number of")

Domain (#Pubs.)	**Type**	**Venue**		**Publications**		
		#Venues	**Sample Venue(s)**	**#Pubs**	**Example Publications**	**Years**
Software Engineering (SE), Programming Languages (PL) & Security	*Conference*	9	ASE, CAV, EuroS&P, FSE, GECCO, ICSE, OOPSLA, TrustCom, ISSTA	28	[27, 51, 68, 81, 82, 103, 134, 196]	2017-21
	Journal	4	EMSE, JSS, RE, TSE	7	[17, 23, 24, 75, 189, 190, 232]	2009-21
	Other	1	ICSE-C	1	[7]	2021
Natural language processing (NLP)	*Conference*	2	ACL, EMNLP	6	[32, 33, 74, 172, 180, 234]	2017-21
Artificial Intelligence (AI) & Machine Learning (ML)	*Conference*	8	AAAI, AISTATS, ICML, NeurIPS, PMLR	17	[5, 124, 149, 225] [4, 40, 111, 120, 168, 213]	2013-21
	Other	1	HRLC	1	[233]	2021
Computer Vision (CV)	*Conference*	2	ICCV, CVPR	3	[122, 207, 208]	2019-20
	Workshop	1	ECCV	1	[221]	2020
Fairness-targets	*Conference*	2	AAAI-AIES, Facct	21	[10, 25, 31, 42, 76, 98, 121, 175, 186]	2017-21
	Workshop	2	FairWare, FATE	8	[18, 35, 57, 59, 107, 204, 212]	2018-19
Big Data, Data Mining (DM), & Knowledge Discovery (KD)	*Conference*	8	DMKD, ECML-PKDD, EDBT, KDD, ICDM, ICEDT, ICMD, LAK	18	[39, 117, 130, 238] [52, 70, 116, 119, 192]	2010-21
	Journal	5	Big Data, Inf. Science, JDIQ, KAIS, SIGMOD-Record	5	[2, 60, 115] [118, 178]	2012-19
	Workshop	2	BSDUC, KDD-XAI	2	[90, 173]	2018-19
Human Factors & Usability	*Conference*	1	CHI	3	[54, 100, 128]	2019-21
	Journal	1	IWC	1	[37]	2016
Others	*Conference*	3	CCCT, VAST, WWW	6	[38, 69, 91, 114, 126]	2009-20
	Journal	6	CACM, DGRP, Scientific-data, SSRN, IBM Journal of R & D	10	[22, 85, 110, 181, 182]	2018-21
	Workshop	1	CEUR Workshop	1	[191]	2019
	Other	3	arXiv, HRDAG, MS Tech. Report	15	[26, 55, 101, 112, 140, 177, 185]	2019-21

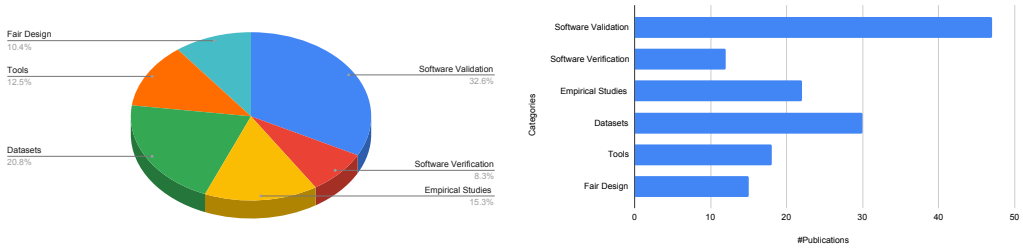
software systems. Secondly, we analyse the fairness measure studied in the papers, such as individual, group, causal or intersectional fairness (RQ2). Next, we study the bias, i.e., the sensitive or protected attributes (e.g., gender or race) studied in the literature (RQ3). We also examine the tasks and datasets employed in the reviewed publications (RQ4). Finally, we investigate the tools available for software fairness analysis in the literature (RQ5).

Specifically, we aim to address the following research questions (RQs).

- **RQ1 Purpose of Fairness Analysis.** What is the purpose of fairness analysis in the literature? What fairness problem is addressed or studied in the literature? What is the target or focus of the community in terms of fairness? What areas have been well-examined or un(der-)investigated in the research community?
- **RQ2 Fairness measure.** What are the fairness metrics analysed by the research community (e.g., individual, group or causal fairness)? What metrics are well-studied, under-studied or not investigated?

Table 2. Purpose of Software Fairness Analysis (recent (2022 and 2023) publications are in bracket)

Categories	Sub-category	Description	#Pubs	Sample Works
Validation	Testing	generating discriminatory test inputs to expose fairness violations	20 (13)	[228, 231] ([11, 45, 133, 214])
	Mitigation	mitigating bias in software systems, e.g., via repair and prevention	14 (13)	[9] ([48, 82, 154, 163, 198, 229])
	Debugging	diagnosis and explanation of fairness violations	8 (3)	[51, 134, 190] ([150, 159]
	Auditing	analysing and measuring bias in software systems	2 (1)	[38, 124] ([230])
Verification	Verifiers	verifying that a system fulfills a fairness metric or goal	12 (1)	[8, 21, 86, 111] ([29])
	Certification	certifying that a system fulfills a fairness goal	4 (1)	[70, 184] ([29])
	Proof or Guarantees	providing a formal proof that a system achieves a fairness goal	2 (2)	[42, 149] ([87, 99])
Design	Requirements	requirement engineering and formalization of fairness properties	4 (2)	[24, 75, 137] ([20, 166])
	Bias-aware Design	designing fair systems and bias-aware software	15 (5)	[39, 107, 116] ([83, 109, 199])
Empirical Evaluation	Analysis	empirical studies about fairness concerns	22 (14)	[28, 35, 59, 234] ([96, 153, 210])
	Benchmarking	providing fair benchmarks or benchmarks for fairness evaluations	4 (3)	[27, 33, 103, 208] ([88, 102, 205])
Datasets	Bias in Datasets	studying biases in training and evaluation datasets	30	[40, 85, 207, 221]
	Bias-aware Datasets	developing unbiased or bias-aware datasets for better evaluation	5 (1)	[35, 173, 180, 181] ([205])
Tooling	Automatic	providing fully automatic tools for fairness analysis	18	[8, 22, 26, 86]
	Semi-automatic	building tools that require human interaction for fairness analysis	2	[38, 136]



(a) Distribution of Publications on Software Fairness

(b) Focus of Software Fairness

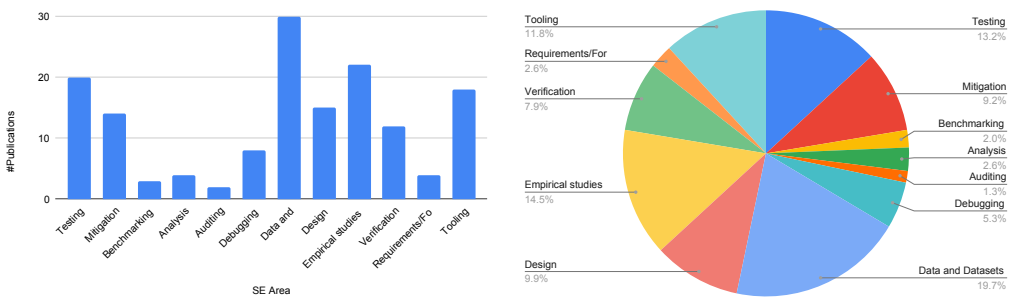
Fig. 14. General Focus of Fairness Analysis

- **RQ3 Bias and Sensitive Attributes.** Which (societal) biases (e.g., age, race gender or religion) are investigated in the analysis of software fairness, especially w.r.t. to protected or sensitive attributes?
- **RQ4 Datasets and Tasks.** What datasets and tasks are employed for software fairness analysis? What tasks/-datasets have been well or under-studied? What is the distribution of datasets per tasks?
- **RQ5 Tooling.** What are the available fairness analysis tool(kit)s, frameworks and libraries? What problems do these tools address, what are the analysis goals supported by available tools? Which analysis approaches are employed in the tools? What stage of model processing do these tools support, and what level of model access do they require?

5 RESEARCH FINDINGS

5.1 RQ1 Purpose of Fairness Analysis.

In this study, we investigate the purpose of each paper collected in this survey, particularly, we examine and categorise the fairness problem studied or addressed by each paper. Table 2 provides details of the purpose of software fairness analysis performed in the literature. Overall, we identified six (6) categories for all collected papers. In the following, we discuss the problem addressed in each category, and the notable works that address such issues. In addition, we highlight the gaps in each category and across all identified categories. Figure 14 and Figure 15 show the the research focus of each paper and the purpose of the fairness analysis conducted in each of these papers, respectively. We also examine how the research community analyzes software fairness, especially the SE, PL and Security venues. We are



(a) Number of Software Fairness works in the SE community

(b) Distribution of Software Fairness Works in the SE community

Fig. 15. Purpose of Fairness Analysis in SE community

interested in the focus of the community, the areas that are well investigated by the community, and the areas that are not explored. Table 3 provides high level details of some of these publications.

Generally, we observed that *the focus of the research community has been on the validation, design and empirical studies of software fairness*. Particularly, one in three (about 33% of) the collected papers study fairness validation, i.e, the testing, debugging and mitigation of fairness errors (*see* Figure 14(a)). Analogously, more than one-third of collected papers (together over 36%) either study biases in datasets (about 21%) or conduct empirical evaluation of software fairness (about 15.3%). Figure 15 shows the focus of the papers published in SE venues. An inspection of these works further show that *dataset analysis, empirical studies, testing and tooling of software fairness is popular* in the community (*see* Figure 15 (a)). In particular, fairness *debugging, requirements analysis, benchmarking, and auditing are not popularly studied in the community* (*see* Figure 15(b)). On one hand, this suggests that the focus of the majority (70%) of the (SE) research work is focused on the validation, empirical evaluation and data(sets) analysis of fairness properties. On the other hand, concerns such as the auditing, debugging and requirements analysis are under-studied.

Most publications (70%) study the validation, empirical evaluation and data(set) analysis of software fairness: The design, verification and tooling of fairness-aware software have been under-studied (about 30%).

Bias Testing, Debugging and Mitigation. The research papers in this category aim to detect, expose, diagnose or mitigate fairness issues in learning-based software systems. Figure 15 provides the distribution for each category. The bulk of this work are general-purpose approaches proposed in the SE, PL and security venues, while the rest of the proposed approaches are more *specialised* approaches proposed for analysing applications in specific domains, e.g., CV or NLP venues. Overall, we found that most approaches are focused on fairness testing, some methods are focused on the mitigation of unfairness, while very few techniques are focused on debugging (diagnosing and understanding the root causes of) unfairness. Indeed, we note that the root cause of unfair program behaviors stem from multiple sources, including societal or historical sources. In this work, we refer to debugging in terms of identifying the software components or configuration that induce the fairness behavior in the software. This notion is inline with previous works [43, 44, 51, 134, 150, 170, 201]. In the following, we shed more light on the main idea of available approaches and the gaps in techniques.

Fairness testing techniques employ a plethora of techniques for test generation, including ML, search and program analysis techniques. Table 3 provides details on the fairness testing methods proposed in the literature. These test

Table 3. Excerpt of works performing Software Fairness Analysis ("Acc." means "level of software access")

Acc.	Approach	Goal	Problem	Main Idea	Core Technique
Black-box	LTDD [134]	Debugging	identifying biased features in training data	debugging biased features to build fair ML software.	data debugging, linear-regression
	Multiacc. Boost [124]	Auditing, Analysis, Mitigation	audit/mitigate multiaccuracy, i.e., group fairness for all subgroups	perform multiaccuracy audit and post-process models to achieve it	multiaccuracy auditing, post-processing
	Cito et al. [51]	Debugging, Mitigation	debug and isolate the cause of mispredictions in ML models	characterize the data on which the model performs poorly	rule induction
	ASTR-AEA [190]	Debugging, Testing, Mitigation	performing fairness testing without existing datasets	discover and diagnose fairness violations in NLP software	grammar-based testing
	Fair-Way [44]	Debugging, Testing, Mitigation	detect and explain how ML model acquires bias from training data	identify how ground truth bias affects ML fairness	multi-objective optimization, pre/in -processing
	FairSM-OTE [43]	Debugging, Testing, Mitigation	finding biased labels in training data generation	remove biased labels and balance data using sensitive attribute	situation testing, data balancing
	Fair-Vis [38]	Auditing, Analysis	auditing and analysing group fairness in ML model	visual analytics for the discovery and audit of (sub)group fairness	visual analytics, domain knowledge
	Flip-test [31]	Testing, Mitigation	testing individual fairness – similar treatment of protected statuses	discover individual (un)fairness and its associated features	optimal transport, flipset, distr. sampling
	Aequ-itas [203]	Testing, Mitigation	validation of fairness for arbitrary ML models?	generating discriminatory inputs to uncover fairness violations	directed testing, probabilistic search
	Themis [12, 36, 81]	Formaliz., Testing	formalize software fairness testing for casual discovering of discrimination	measure causal discrimination in software to direct fairness testing	input schema, causal relationships
	Aeque-Vox [170]	Debugging, Testing	testing group fairness for Automatic Speech Recognition (ASR) systems	group fairness testing by simulating different environments	ML robustness, test simulation, fault localization
	ExpGA [68]	Testing	current individual fairness testing methods suffer poor efficiency, effectiveness, and model specificity	fairness testing by modifying feature values using explanation results and genetic algorithm (GA)	genetic algorithm, feature mutation, search based testing
	CGFT [151]	Testing	Uneven distribution of fairness tests and variations in execution results	leverage combinatorial testing to generate evenly-distributed test suites	combinatorial testing, input coverage
	SG [6]	Testing	detecting the presence of individual discrimination in ML models.	auto-generation of test inputs for detecting individual discrimination.	symbolic execution, local explainability
	Bias-Finder [17]	Testing	Bias testers for SA systems rely on small, short, predefined templates	discover biased predictions in SA systems via metamorphic testing.	template curation, NLP techniques, metamorphic testing
	Biswas and Rajan [27]	Mitigation	understanding fairness characteristics in ML models from practice	empirical evaluation of fairness and mitigations on real-world ML models	empirical study
	Fairea [103]	Mitigation	what is the SE trade-off between accuracy and fairness?	benchmarking and quantifying the fairness-accuracy trade-off achieved by bias mitigation methods	model behaviour mutation
White-box	ADF [231, 232]	Testing	searching individual discriminatory instances	generating discriminatory inputs violating individual fairness via ML	gradient computation and clustering
	Deep-Inspect [200]	Testing	detecting confusion and bias errors at class-level	expose confusion and bias errors in image classifiers	class property violations, robustness
	EIDIG [228]	Testing, Mitigation	how to detect and improve individual fairness of a model	generating test cases that violate individual fairness	gradient descent, global/local search
	Neuron-Fair [235]	Testing, Mitigation, Analysis	interpretability, performance, and generalizability in bias testing	identifying biased neurons, i.e., neurons that cause discrimination	neuron activation, adversarial attacks
	Fair-Neuron [82]	Mitigation, Analysis	balancing accuracy-fairness trade-off without additional model(s)	detect neurons with contradictory optimization directions, and achieve trade-off via selective dropout	joint-optimization, adversarial game
Grey	Tizpaz-Niari [201]	Debugging, Testing, Mitigation	explaining fairness impact of hyper-parameters	identify the effect of parameters on software fairness	search based testing, statistical debugging
	CAT/ TransRepair [195, 196]	Testing, Mitigation	detecting inconsistency in machine translation (MT)	detect inconsistency bugs without access to human oracles	mutation testing, metamorphic testing, language model (BERT)

generation methods include several black box testing approaches (especially, input-based approaches), and few white-box

techniques. Notably, *there are even fewer grey-box fairness testing approach*. Thus, most proposed approaches drive the generation of discriminatory inputs either by *analysing the model under test (MUT)* or the *input space*. However, there are very few techniques that leverage both the input space and the model analysis, besides, there are few studies investigating the relationships between both dimensions for testing purposes (e.g., Tizpaz-Niari et al. [201]). Notably, white box approaches (e.g., ADF [231, 232] and EIDIG [228]) mostly employ ML techniques (e.g., gradient computation, and clustering) to drive the generation of discriminatory test cases. Meanwhile, black-box approaches focus on leveraging the knowledge of the input space, program analysis and/or search algorithms to generate discriminatory inputs. They mostly employ templates, schemas, grammar, mutation or search algorithms to drive fairness test generation [190, 195, 203, 219]. Other approaches employ program analysis, e.g., symbolic execution [6] and combinatorial testing [151] to drive the generation of discriminatory test inputs. Notably, *we found few grey-box testing techniques that leverage both the input space and internal model attributes/properties to drive the generation of discriminatory inputs*. Moreover, there is little work that studies the link between the properties of the input space or discriminatory inputs to other internal model attributes/properties for fairness testing (e.g., Tizpaz-Niari et al. [201]).

Fairness testing approaches are mostly black or white box test generation methods: There are few grey-box approaches that leverage (or study) the relationship between the input space and internal model properties.

Fairness Verification, Certification and Proof Guarantees: We examine the literature on the verification, certification and proving of fairness properties in learning-based software systems. Specifically, we examine the categories of verifiers, the main ideas of proposed verification approaches and the gaps in this research area. To this end, we identify three major kinds of fairness verification approaches, namely *distributional verifiers* (e.g., FairSquare [8] and VeriFair [21]), *specialised verifiers* designed for a particular domain, metric or task (e.g., [42, 70, 111, 138, 143]) and sample-based verifiers (e.g., Themis [81]). In our study, most approaches are *distributional-based* verifiers or *specialised* verification techniques, and there are fewer *sample-based verifiers*. Distributional verifiers typically encode fairness metrics as probabilistic properties then verify such properties with respect to the underlying data distribution of the learning-based system. On the other hand, sample-based approaches (e.g., Themis [81]) allows to detect and verify fairness metrics based on a fixed dataset, otherwise they generate counter-examples to refute the satisfaction of the fairness property. Other verification approaches target specific domains (e.g. NLP [143]), specific fairness metrics (e.g., individual fairness [111] and disparate impact [70]), or tasks (fair training [42] and data debiasing [70]).

Some verification approaches encode fairness metrics as probabilistic properties, then provide guarantees over the system's data distribution. These approaches take as input the probability distribution of the attributes in the dataset and the MUT, then verify the fairness of the system with respect to the distribution and MUT. For instance, FairSquare [8] presents a technique for verifying and certifying fairness properties by encoding fairness definitions as probabilistic properties. Moreover, Albarghouthi et al. [9] also proposed an approach that applies *distribution-guided inductive synthesis* to verify (and repair) *unfair* ML classifiers. Likewise, Bastani et al. [21] developed an algorithm for verifying fairness specifications (called VeriFair), the algorithm provides probabilistic guarantees for fairness properties and allows users to verify that the probability of fairness errors is small. Ghosh et al. [86] presents a stochastic satisfiability (SSAT) framework (called Justicia) to formally verify fairness measures of supervised learning algorithms with respect to the underlying data distribution. Justicia is applicable to different fairness metrics including disparate impact, statistical parity, and equalized odds. Compared to previous distribution-based verification approaches (FairSquare [8] and VeriFair [21]), Justicia supports non-Boolean and compound sensitive attribute. It also provides theoretical bound for the finite-sample error of the verified fairness measure, and it is more robust than sample-based verifiers.

There are several verification approaches that are focused on a specific domain, metric, or task (e.g., debiasing datasets [70] or fair training [42]). For domain- or task-specific approaches, Ma et al. [143] provides a black-box technique to enforce fairness guarantee for NLP systems by leveraging advances in certified robustness of machine learning. Their approach employs a neutral phase to piggyback the NLP model to smooth its outputs such that they are certified to preserve individual fairness. Similarly, Liu et al. [138] presents a (semi-)automated verification framework (called FairCon) to ascertain the fairness of smart contracts, their approach can refute false claims with concrete examples, or certify that contract implementation fulfil desired fairness properties.

For metric-specific approaches, John et al. [111] proposes sound (but incomplete) verifiers for proving individual fairness of models by employing appropriate relaxations of the problem, specifically for linear classifiers and kernelized polynomial/radial basis function classifiers. Likewise, Feldman et al. [70] presents a verification technique for certifying the (im)possibility of disparate impact on a data set by employing a regression algorithm that minimizes the balanced error rate (BER) of the dataset. The goal is to verify that a protected or sensitive attribute can not be predicted from the other attributes in the dataset by ascertaining if there is sufficient information about the dataset to detect sensitive attributes from the data. In terms of verifying fair training, Celis et al. [42] propose a technique to train fair classifiers with theoretical guarantees, using a meta-algorithm for classification that can take as input a general class of fairness constraints with respect to multiple non-disjoint and multi-valued sensitive attributes. Notably, this approach can handle non-convex fairness constraints such as predictive parity.

A different line of fairness verification techniques are sample-based verifiers such as AIF360 [22] and Themis [81]. Typically, these approaches leverage software testing techniques to verify fairness properties on fixed data sample. AIF360 [22] is an extensible open source toolkit for detecting and verifying fairness properties, particularly for a fixed data sample. It provides an array of methods to detect and report several fairness performance metrics. This includes 71 bias detection metrics, and nine bias mitigation algorithms. Meanwhile, Themis [81] allows developers to verify that a fixed sample do not discriminate against specific sensitive attributes (e.g., race and gender) by automatically generating discriminatory that verifies that changing the instance of the attribute does not cause a change in the output of the learning-based system. Overall, these approaches leverage advances in software testing to measure and verify that a fixed sample data fulfills specific fairness properties.

There are other general verification approaches that verify multiple (user-defined) fairness constraints or other properties beyond, but including, fairness properties. As an example, Metevier et al. [149] addressed the problem of verifying multiple fairness definitions as well as user-defined fairness metrics for learning-based systems. The authors proposed a verification approach (called RobinHood) which employs an offline contextual bandit algorithm determine the satisfiability of several fairness constraints. Morevoer, RobinHood provides a probabilisitic guarantee of fairness by ensuring that it does not return a solution with a probability greater than a user-defined threshold. Besides, Sharma et al. [185] is a more general verification approach which is also applicable to fairness properties. The authors propose a (white-box) verification approach that employs the knowledge of the internal structure of the model to verify that ML models fulfil several properties (including fairness), in particular by training a shadow model that approximates the MUT by using the prediction of the original model as training data. It employs a property specification language to test and verify model properties of learning-based software and provides counter-examples (i.e., test cases violating the property) if the property is not fulfilled.

Most fairness verifiers are distribution-based, sample-based or specialised for a specific fairness measure, domain or task. There are very few verifiers that support multiple or user-defined fairness constraints.

Others: Our investigation showed that the SE research community has *mostly* focused on analysing software fairness as a *fairness validation (i.e., testing and debugging)* problem [81, 150, 203] and as a *fair system design* problem, especially to mitigate biases [37, 44, 202]. However, other aspects of SE concerns are under-studied, such as the formalization of fairness as a *software requirement* [36, 75], the *verification* of fairness properties [8], and *empirical evaluation* of software fairness properties [27, 103]. Furthermore, some aspects of SE concerns are hardly studied by the community, namely, empirical evaluation of fairness properties (especially human factors in software fairness [73, 79, 91, 156]) and the maintenance of fairness properties as the software evolves (e.g., because of model re-training, model compression [101] or software regression).

In 2017, Brun and Meliou [36] formalised software fairness as a software engineering problem that needs to be tackled by all aspects of software engineering. These aspects include steps in the software development life cycle such as requirements engineering, design, testing, verification and maintenance. Since their publication, the bulk of the published papers have *focused on the design, testing and mitigation* of software fairness, these areas have been well explored and investigated by these communities. For instance, several papers have explored the problem of *fairness testing* by employing random test generation (Themis) [12], local search algorithms (Aequitas) [203], gradient computation (ADF and EIDIG) [228, 231, 232], mutation testing (TransRepair) [195], grammar-based testing (Astraea) [190], symbolic execution [6], property-driven testing [185] and schema or template based testing (BiasRV) [219]. In addition, the problem of fair system design to mitigate biases has been studied by a few researchers via several techniques including behavior mutation [103], and feature or dataset manipulation [226]. Researchers have also proposed search-based optimization methods for the fairness-aware design, testing and mitigation. For instance, Perera et al. [164] proposed a search-based method for regression fairness testing and Hort et al. [104] presents a search-based method for the repair of fairness and accuracy properties of ML-based software. Hort et al. [102] have proposed a multi-objective search method to determine a balance between gender-fairness and semantic correctness of word embeddings using Word2Vec.

Very few researchers have conducted empirical evaluation of software fairness properties in real-world applications. Notably, Biswas and Rajan [27] conducted an empirical study to study several fairness mitigation techniques including their impact on performance. Likewise, Hort et al. [103] conducted a large scale empirical study to test the effectiveness of 12 widely-studied bias mitigation methods. Meanwhile, Zhang and Harman [226] empirically evaluated how feature set and training data affect fairness. The focus of most studies has been on fairness mitigation, except Zhang and Harman [226] that empirically studied the impact of features and datasets on fairness properties. Besides these two concerns, we have found few studies in these fields empirically evaluating other SE concerns (such as human factors [54, 79, 92, 100, 156]) or concerns relevant to steps of the model/software development pipeline.

However, some aspects of software fairness analysis remain under-investigated. Firstly, there is little work addressing concerns about the maintenance of software fairness, for instance, as the software or model evolves over time (i.e., software regression analysis) or is optimized (e.g., via model compression for edge devices). For instance, there is a recent paper investigating the impact of model compression on software fairness [101]. We also found few works supporting the software engineering activities around such changes. Notably, fairness testing for changes in ML-based software, i.e., in a regression scenario has been examined by Perera et al. [164]. The authors proposed a search-based approach to address regression in ML-based systems. Likewise, there are very few empirical studies on software fairness properties, particularly, there has been few empirical evaluation in this area involving humans, which is vital to determine the harm caused by unfair software behavior [54, 100] and practitioners (developers) perception of software fairness properties [73]. Secondly, the formalization and definition of fairness as a software requirement, metric or

measure has only been performed by a few researchers. We found few papers in this area including Brun and Meliou [36], Ferrara et al. [72], Verma and Rubin [204] and Finkelstein et al. [75].

The SE research community mostly study software fairness properties as mitigation, design and testing problems. Very few works have studied fairness as a requirements engineering or verification problem, and fewer works empirically studying human factors of fairness properties and how to maintain fairness as software changes/evolves.

5.2 RQ2 Fairness measure.

In this research question (RQ2), we examine the fairness metrics analyzed by the studies and methods proposed for software fairness analysis. We investigate the number of studies examining the different classes of fairness metrics (e.g., statistical measures, similarity-based measures and time-based measures), as well as the distribution of specific metrics, such as individual fairness, group fairness and causal fairness. Figure 16 highlights our analysis of the distribution of these metrics across proposed methods and conducted studies.

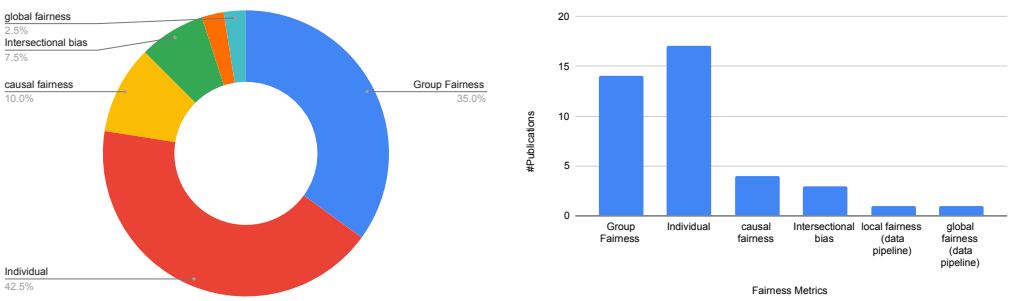
Generally, we observed that most studies examine statistical or similarity based measures (such as individual, group or causal fairness), while time-based metrics (e.g., sequential or long-term fairness) or measures based on causal reasoning (e.g., fair inference) are not (yet) studied in the SE community as software fairness metrics. Statistical and similarity based fairness metrics (such as group, individual, and intersectional fairness) are the most examined metrics in software engineering (*see* Figure 16(a)). For instance, individual fairness and group fairness account for more than three in four (76.5%) studies and methods in software fairness analysis. Causal reasoning based fairness metrics (such as causal fairness) are also commonly studied. However, time-based metrics were not found in the SE literature [233].

Statistical and similarity -based fairness metrics are popularly studied (86.5%) in fairness analysis, but metrics hinged on causal reasoning are under-studied (about 10%) and we could not a find a single work studying time-based metrics.

In addition, we observed that fairness metrics such as *individual and group fairness metrics are well studied in the SE community*. Concretely, about 42.5% and 35% of publications in software engineering venues study individual fairness and group fairness, respectively. The focus of most of these studies is in terms of testing, validating and mitigating software to fulfill these metrics.

Several of these techniques are tailored towards testing, discovering and mitigating one or both of these metrics. Concretely, 27% of examined studies study only individual discrimination [203, 228, 231]. For instance, Zhang et al. [228] proposed EIDIG (Efficient Individual Discriminatory Instances Generator), a scalable and efficient approach to systematically generate test cases that violate the individual fairness for DNN models. Likewise, Zhang et al. [231] proposed approaches to search for individual discriminatory instances of DNN, using lightweight procedures like gradient computation and clustering. Overall, we found that 23% of examined papers in the community study both individual and group fairness, 17% examine only group fairness and 27% study only individual fairness.

In our analysis, causal fairness is also a commonly studied fairness measure in the community, it accounts for one in ten publications. Notably, Galhotra et al. [81] and Biswas and Rajan [28] have studied the testing and debugging of causal fairness. Meanwhile, other metrics such as intersectional bias, local and group fairness (especially in ML or data pipelines) are the least studied in the community, with each accounting for less 2.5 to 7.5 percent of all examined works. As an example, Chakraborty et al. [43], [124] and Cabrera et al. [38] studied the analysis of intersectional bias in SE venues, specifically, in terms of testing, auditing and visual analysis of intersectional bias, respectively.



(a) Distribution of studied Fairness Measures

(b) Number of Publications per fairness measures

Fig. 16. Details of Fairness Measures

A recent survey by Gohar and Cheng [89] examines the state-of-the-art in intersectional fairness and presents a taxonomy for intersectional notions of fairness and mitigation. Overall, there are still several open issues in this area. Particularly, how to effectively test, mitigate and debug intersectional software bias and how intersectional bias interacts with other fairness metrics (e.g., group and individual fairness).

We also examine the composition, sequential and long-term aspects of fairness properties. Notably, Gohar et al. [88] have recently studied fairness composition in ensemble models. The authors show that ensembles can be designed to be fairer without using mitigation techniques and provide a benchmark for studying fair ensembles. Analogously, Chen et al. [47] proposed an ensemble method (called MAAT) for improving fairness-performance trade-off for ML software. MAAT combines models optimized for the two objectives – fairness and ML performance. Finally, on the extreme end, we found almost no work in the SE community studying time-based fairness metrics such as sequential and long-term fairness [183, 233]. These metrics are important because of maintaining fairness properties as the software evolves. As an example, consider an automated classifier that is re-trained periodically due to new data requirements, how do we ensure that such data and the resulting classifier maintains the fairness property defined in the previous system? Overall, this shows that the focus of the community is on a set of statistical and similarity-based fairness metrics (such as individual, group, causal), hence, ignoring other metrics, e.g., time related bias concerns.

Most SE studies (86.5%) investigate individual fairness, group fairness and causal fairness metrics, metrics such as intersectional fairness and time-based metrics (sequential and long-term fairness) are under-studied in the SE community.

5.3 RQ3 Bias and Sensitive Attributes.

Let us investigate the societal biases studied in software fairness. In particular, we analyze the sensitive or protected attributes (e.g., age, race, gender etc) that researchers study or develop techniques for. Figure 16 and Figure 17 illustrates the distribution of sensitive attributes in the literature. In Table 4, we exemplify and illustrate some of these sensitive attributes. We note that the examples in Table 4 are not real violations. However, they are adapted from fairness violations discovered by state-of-the-art NLP bias testing tools [17, 143, 172, 190].

We observed that *about four in every five studies examine four main protected attributes, namely age, race, gender and country*. These four attributes are studied in about 79.5% of works, with age, race and gender accounting for the majority (about 72.5% of all works). We believe this is due to the availability of these attributes in several datasets. For

Table 4. Illustration of studied biases, i.e., sensitive/protective attributes. These examples are adapted fairness violations discovered by state-of-the-art bias testing tools [17, 143, 172, 190]. Sentiment analysis (SA) systems detect the emotional situation or state in a sentence, ☺ indicates a positive sentiment (e.g., happy),☹ indicates a negative sentiment (e.g., sad), bias-inducing inputs are underlined and unfair outputs are in red (e.g. ☹).

Sensitive Attributes	**#Pubs**	**Illustrative Example for text-based SA system**	**Sample Works**
Gender	24	"The {man/woman} is happy." = ☺/☹	[43, 44, 203]
Race	21	"The {white man/ black man} is happy." = ☺/☹	[81, 231, 232]
Age	15	"The {young man/ old man} is happy." = ☺/☹	[27, 226, 228]
Country	6	"The {american man/ chinese man} is happy." = ☺/☹	[170, 232]
Language	2	"The {english orator/ spanish orator} is happy." = ☺/☹	[195, 196]
Occupation	2	"The {manager/ cleaner} is happy." = ☺/☹	[17, 190]
Religion	2	"The {nun/ atheist} is happy." = ☺/☹	[190, 232]
Ethnicity/Accents	2	"The {european man/ asian man} is happy." = ☺/☹	[170, 232]
Class label (e.g., poverty level)	1	"The {rich man/ poor man} is happy." = ☺/☹	[43, 200]
Narcotics Arrests /Gang Affiliation	1	"The {convict/ innocent man} is happy." = ☺/☹	[31]
Work experience	1	"The {experienced farmer/ novice farmer} is happy." = ☺/☹	[31]
Academics (*LSAT /GPA*)	1	"The {honors student/ failing student} is happy." = ☺/☹	[31]
Marital-status/Relationship	1	"The {married man/ single man} is happy." = ☺/☹	[38]

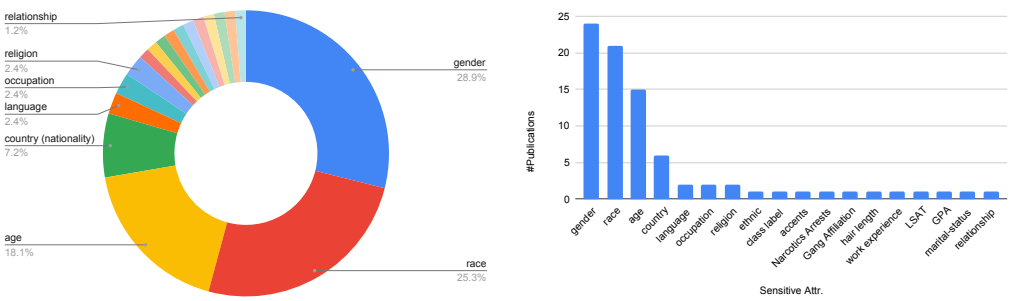
instance, attributes such as age, gender and race are popular features in tabular datasets (e.g., German Credit, Adult Census Income and Bank Marketing). These attributes are popularly studied due to the simplicity of manipulating them. They typically have a tractable input space, bounded constraints or limited range of values. As an example, it is easy to manipulate gender features in tabular data, since they have a small number of possible values (e.g., male, female or non-binary). Meanwhile, rare protected attributes (e.g., hair length [137] or narcotics arrest, [157], accents [160, 211] and language features [58, 139]) either possess a complex input space, are uncommon in popular tabular datasets, or are more relevant to semi-structured datasets (e.g., text, audio and images). Such attributes are more difficult to study or manipulate: For instance, consider an image dataset where the gender-related sensitive attribute are related to pixel values [133]. Despite the complexity of rare attributes, some researchers have explored their fairness concerns. Notably, Black et al. [31] and Lipton et al. [137] both employed hair length, work experience and academic performance in their works for fairness analysis. More recently, researchers have also explored fairness concerns in speech tone [220]. Overall, this analysis suggests that there is a gap in fairness analysis of rare sensitivie attributes. Indeed, the focus of the community has been on the sensitive attributes that are available in less complex tasks or tabular datasets.

About four in five (79.5%) works study age, gender or race as sensitive attributes, while model class-label and specific attributes such as relationship (status), class (academics, work) and religion are understudied (less than 2.5% each).

5.4 RQ4 Datasets and Tasks.

For this research question, we examine the datasets examined or employed in the collected publications. Table 5 provides details of the tasks and datasets employed in (the evaluation of) software fairness analysis. Furthermore, Figure 18, Figure 19 and Figure 22 highlight the type of the dataset (e.g., tabular, or semi-structured), the distribution of examined datasets, and the task associated with each dataset, respectively. We also examine the task category associated with the examined datasets (e.g., NLP, CV , etc), the volume of publications associated with each task category, and dataset (*see* Figure 20, Figure 23 and Figure 24, respectively).

Volume of Publications per dataset: We found that *more than half of the examined studies employ four (4) major datasets, most of which are tabular datasets for finance-related tasks.* Figure 18 illustrates the distribution of publications using each dataset. Notably, the most common datasets are the Adult Census Income, German Credit, Bank Marketing



(a) Distribution of studied Sensitive Attributes

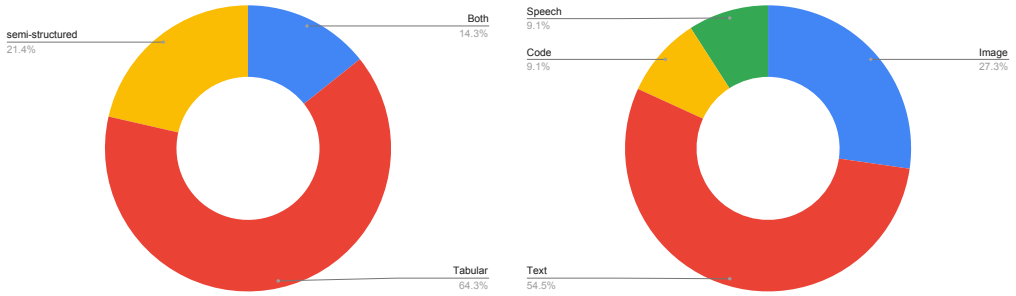
(b) Number of Publications per Sensitive Attribute

Fig. 17. Details of Biases, i.e., Sensitive/Protected attributes

Table 5. Details of Tasks and Datasets employed in Software Fairness Analysis

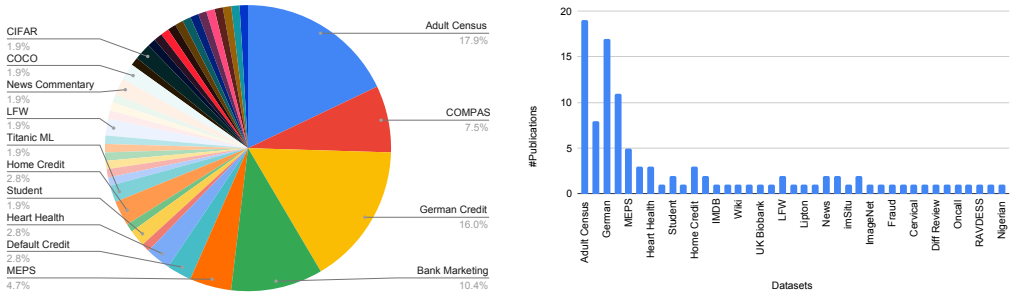
Type	Task Category	#Data.	#Pubs	#Tasks	Tasks (Example Pubs)	Datasets (#Pubs)
Tabular	Education	2	3	1	academic performance [31]	Law School (1)
	Finance	6	54	4	income [6, 226, 231],	Adult Census (19)
					credit default [12, 28, 44]	German Credit (17), Default Credit (3), Home Credit (3)
					potential buyers [28, 43, 134]	Bank Marketing (11)
					fraud [6]	Fraud Detection (1)
	Legal	3	10	3	recidivism [38, 44, 103]	COMPAS (8)
					arrests [31]	Chicago Strategic Subject List (SSL) (1)
					US executions [6]	US Executions (1)
	Medical	5	11	5	medical expenditure [43, 134, 235]	MEPS (5)
					heart disease [43, 44]	Heart Health (3)
					heart failure [51]	Heart Failure (1)
					cancer risk [51]	Cervical Cancer (1)
	Other (HR)	1	1	1	hiring [31]	Lipton (1)
	Other (rentals)	1	1	1	car rentals [6]	Raw Car Rentals (1)
	Other (shipwreck)	1	2	1	shipwreck survival	Titanic ML (2)
Semi-structured	CV (image)	7	10	2	face detection	ClbA-IN (1), PPB (1), LFW (2)
					image recognition	COCO (2), imSitu (1), CIFAR (2), ImageNet (1)
	NLP (text, speech)	12	10	6	SA [17], CoRef [190], MLM [190]	Twitter (1), IMDB (1), EEC Dataset (1), Labor statistics (1)
					toxicity	Wiki Comment (1), Jigsaw Comments (1)
					machine translation	News Commentary (2)
					ASR [170]	Speech Accent Archive (1), RAVDESS (1), Multi speaker Corpora of the English Accents in the British Isles (1), Nigerian English speech dataset (1)
	SE (code)	4	4	4	Programs [51]	Bug2Commit (1), Diff Review (1), Code AutoComplete (1), Oncall Recommendation (1)

and COMPAS dataset, they are employed in over half (52%) of all (SE) papers. These datasets account for most of the software fairness works we examined (*see* Figure 18 (b)). Most of these datasets are tabular datasets for finance-related tasks such as income prediction, except COMPAS – a popular legal dataset for recidivism. However, semi-structured datasets such as image (COCO, CIFAR) and text/NLP (News Commentary) are among the least studied datasets. This suggests that most research work employ similar datasets, which are mostly tabular or finance related. Hence, implying there is the need for a more diverse evaluation of fairness analysis techniques that are general and cuts across several datasets and tasks.



(a) Proportion of Tabular or Semi-structured Datasets (b) Distribution of Semi-structured Datasets

Fig. 18. Type of Studied Datasets

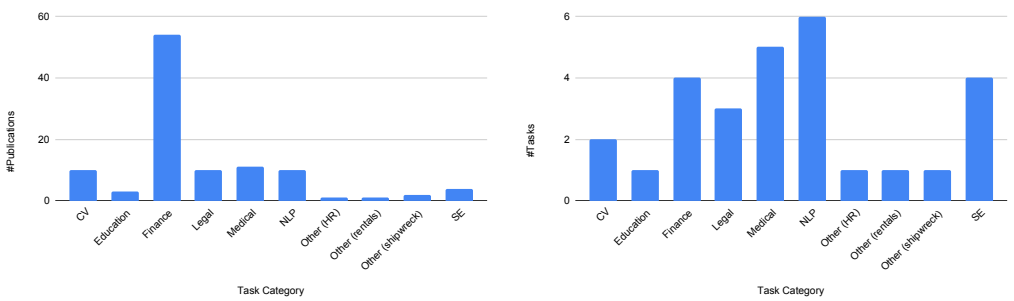


(a) Distribution of examined datasets (b) Number of works (pubs) evaluating a specific dataset

Fig. 19. Details of examined Datasets

Tabular datasets (e.g., Adult Census Income) are the most employed datasets in software fairness analysis. Semi-structured datasets (e.g., image, text, code and speech) are less popularly studied.

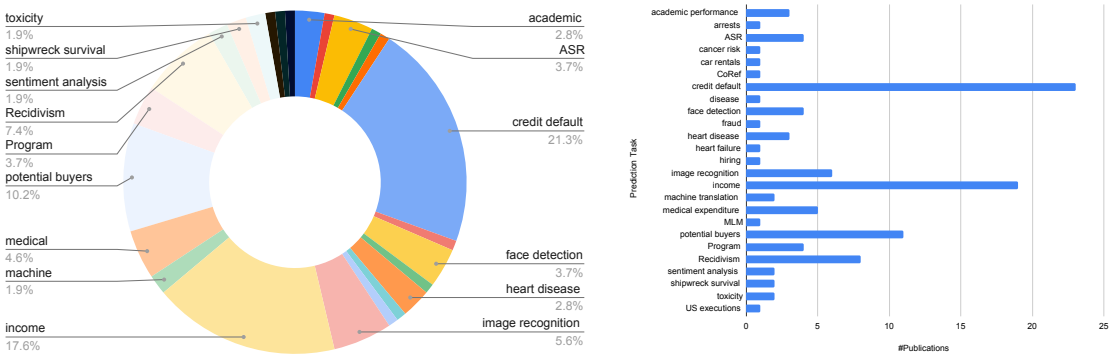
Type of examined datasets: We observed that *the majority (64%) of the research works study software fairness for tabular datasets*. Figure 18(a) shows that about two-third of the examined papers study fairness properties relating to tabular datasets (e.g., Adult Income Census). *Research works that generalise to both tabular and semi-structured datasets are few (about 14%). Fewer studies (about 21% of papers) solely analyse fairness properties for semi-structured dataset* (e.g., text, image, speech or code). Figure 18(b) provides the distribution of works studying semi-structured datasets. Inspecting works studying fairness properties in semi-structured datasets, we observed that most (about 72%) are focused on text (i.e., NLP-related) datasets or image (i.e., CV) datasets. *Datasets relating to speech (i.e., audio) and code (i.e., programs) are the least studied in software fairness*, accounting for about 18% of the examined papers. Generally, the focus of the community has been on tabular datasets. Overall, this suggests that the SE research community has been focused on studying fairness properties as related to tabular (relatively less complex) datasets, while more semi-structured datasets are under-explored. This suggests the need to develop fairness analysis techniques that are general and agnostic, i.e., applicable to both tabular and semi-structured datasets.



(a) Number of works (pubs) examining a task type

(b) Number of tasks studied in each category

Fig. 20. Details of Task Categories



(a) Distribution of Publications across tasks

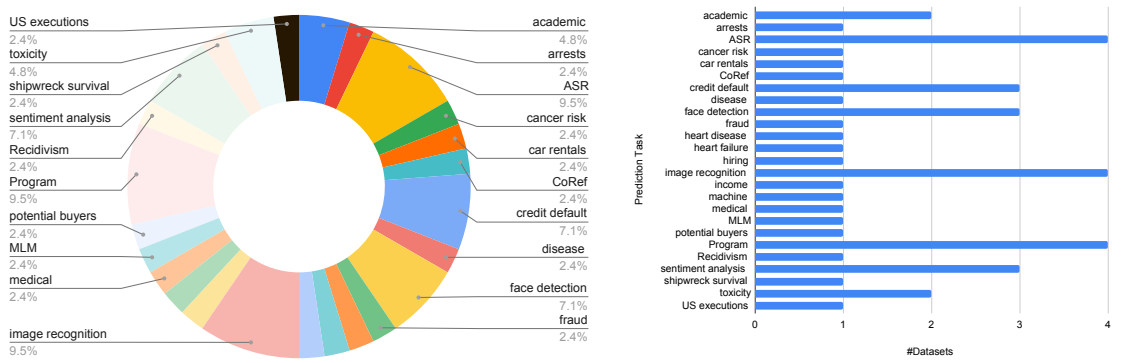
(b) Number of Publications across tasks

Fig. 21. Details of Publication across tasks

Researchers have recently taken interest in analyzing fairness properties in program-based, and SE-task centric activities. For example, researchers have examined fairness properties for SE-centric tasks such as issue assignment [153], buggy commit classification [51], code review recommendation [145], crowd worker testing recommendation [206], automatic code generation [106] and software development pipelines [96, 210]. Similarly, there has been an increasing attention to bias analysis of LLMs [1, 106, 179], especially due to their importance and recent popularity.

Fairness concerns relating to semi-structured datasets such as code and speech (audio) are rarely studied, they account for only about 18% of all inspected papers.

Task Categories: Figure 20 and Figure 22 highlight the details of the task categories and the datasets employed for each task (category). We found that *tasks involving finance, computer vision (CV) and natural language process (NLP) are the most studied in software fairness publications, both in terms of the number of tasks and the number of publications.* However, software fairness concerns for tasks involving *education* and *code analysis* are less frequently studied. Figure 20 illustrates that all of the well studied task categories (CV, NLP and finance) have up to two to four tasks each, while education had one task. As an example, finance-related publications contribute over 50 papers (*see Figure 20(a)*), with



(a) Distribution examined tasks

(b) Number of datasets per task

Fig. 22. Details of Datasets examined for each Task

about four distinct tasks examined (Figure 20(b)). These tasks include predicting income, credit default, fraud and potential buyers (Figure 22). In addition, despite fewer publications, we observed that some task categories have more examined tasks. For instance, consider NLP task category, over six tasks have been studied in the literature, e.g., sentiment analysis, CoRef, MLM, ASR etc. This is despite very few publications for fairness analysis of NLP systems.

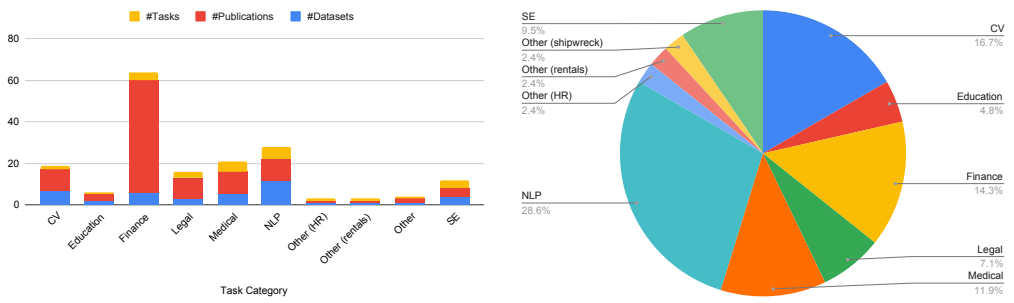
Our analysis of the tasks investigated in the collected papers confirm that *financial, CV and NLP tasks are well studied for software fairness analysis*. Figure 23 and Figure 24 further shows the distribution of publications based on the number of tasks, and datasets. Similarly, Figure 21 shows that finance related tasks are the most studied, especially tasks such as credit prediction, income prediction and fraud detection. Figure 22 provides more fine grained analysis of the datasets relating to each task. Evidently, tasks involving finance, CV and NLP account for most examined datasets, about seven (7) to nine (9) percent each. For instance, tasks involving face detection, credit default, image recognition, speech recognition (ASR) and programs contribute about four (4) datasets each (*see* Figure 22(b)). These findings suggests that the community has been focused on investigating bias in specific sectors (CV, NLP and finance) while ignoring other categories (e.g., education and code).

The (SE) research community has been focused on studying software fairness concerns for (three) specific tasks (finance, CV and NLP tasks), but fairness concerns in areas like education and code analysis have been largely ignored.

5.5 RQ5 Tooling

In this evaluation, we inspect the papers that propose a tool, framework or library to enable software fairness analysis. Table 6 provides details of some of the tools found in the literature. We categorize the goal of the analysis performed by each tool, the addressed problem, the main approach employed and their processing stage (pre, in and post -processing), as well as the software access required by the tool (black, grey or white box access).

Our analysis showed that *post-deployment validation of AI software (e.g.. testing, auditing, analysis and mitigation) is the most prominent tool support available for software fairness analysis*. Table 6 shows that most tools are black-box post-validation tools. This is followed by support for fair learning (i.e., designing fair software systems). These includes measuring and analyzing the trade-off between fairness and accuracy metrics of the software, e.g., AIF360 [22] and



(a) Distribution of examined datasets

(b) Distribution of the Number of tasks per Task Category

Fig. 23. Details of Publications and Datasets for each Task category

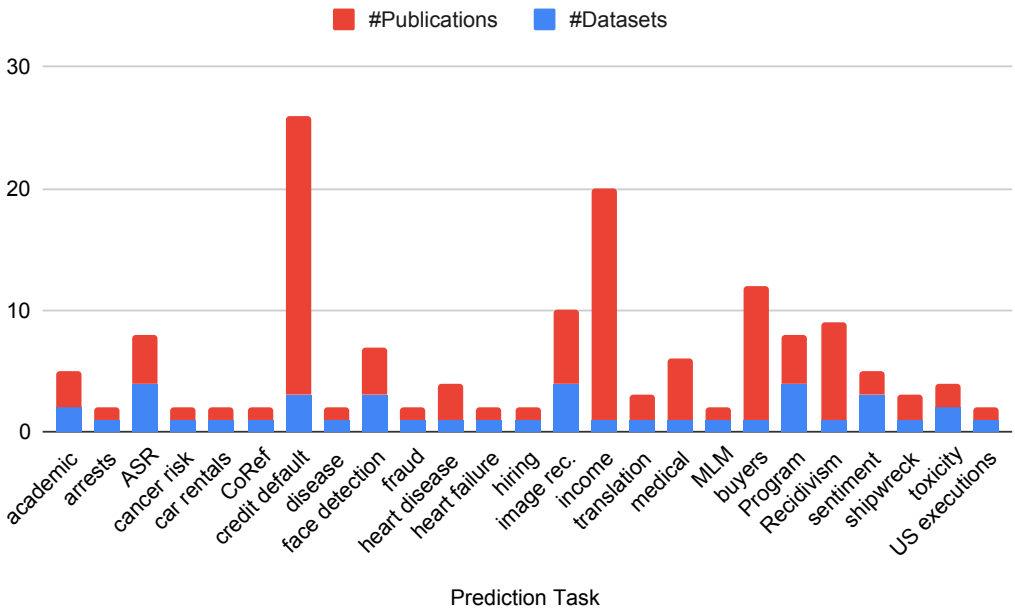


Fig. 24. Details of Publications and Datasets per Prediction Task

POF [25]. Overall, *there is very little tool support for the specification, formalization and verification of fairness metrics.* Some tools support the verification of fairness properties in the post-processing stage, i.e., verifying trained model (e.g., FairSquare [8]). Meanwhile tools like VeriFair [21] verify fairness properties in the pre-processing stage, i.e., verifying if datasets fulfill a fairness property. There is also *little support for in-processing and white-box analysis of software fairness.* While there are some works that support all processing stages (i.e., pre, in and post) of software fairness [19, 22, 112, 113, 189], we found low support for tools specifically focused on the in-processing stage.

Table 6. Excerpt of Fairness Analysis Tools

Tool (Paper)	**Goal**	**Addressed Problem**	**Process. Stage**	**Approach**	**Access**
Fairkit-learn [112, 113]	Fair learning, Analysis	how to reason about and determine the trade-off between model quality (accuracy) and fairness	pre, & post	model search, visualisation	Grey
AIF360 [22]	Fair learning, Analysis	understanding how, when and why to use different bias handling algorithms in the model life-cycle	pre, in, & post	extensible architecture for analysing fairness metrics	Grey
POF [25]	Fair learning, Analysis	how to compute the "Pareto curve" of the trade-off between accuracy and fairness in the *regression* settings (*continuous* prediction/targeted values)	pre	fairness regularizers, Price of Fairness (PoF) metric	Grey
AITEST [7]	Testing	how to detect the presence of individual discrimination in ML models	post	symbolic execution and local explainability	Black
2AFC [136]	Testing	how to relate unobservable phenomena deep inside models with observable, outside quantities that we can measure from inputs and outputs	post	Test Experiments, Experimental Psychology, Psycophysics, two-alternative forced choice (2AFC)	Black
Pc-fairness [213]	Formalization, Analysis	how to bound path-specific counterfactual fairness, address their *identifiability*, i.e., whether they can be uniquely measured from observational data	post	parameterized causal modelling, linear programming, response-function variables, constraints	Black
BiasRV [219]	Testing	how to monitor and uncover biased predictions at runtime	post	automatic template generation, mutation, metamorphic relations	Black
Themis [12]	Formalization, Testing	how to formally define and test software fairness using a causality-based measure of discrimination	post	causal inference, schema -based test generation	Black
FairSquare [8]	Formalization, Verification, Certification	how to verify or certify that a program meets a given fairness property	post	probabilistic reasoning, SMT solving, symbolic weighted -volume-computation algorithm	Black
FAT Forensics [189]	Analysis, Auditing, certifying	how to inspect datasets (features), models and their prediction for fairness metrics	pre, in, & post	an inter-operable Python framework for fairness (FAT) algorithms	White, Black, & Grey
VeriFair [21]	Verification, Specification	how to verify fairness specifications, i.e., fairness properties of ML programs	post	adaptive concentration inequalities	Black
Justicia [86]	Verification	how to formally verify the fairness metrics are satisfied by different algorithms on different datasets	pre	stochastic satisfiability (SSAT)	Black
Checklist [172]	Testing	how to test (fairness) behaviors of NLP systems	post	behavioral testing, template -based test generation	Black
FairML [3]	Auditing, Analysis	how to determine the significance of inputs in assessing the fairness of black-box models	post	model compression, input ranking algorithms	Black
MT-NLP [143]	Testing, Mitigation	how to determine if NLP models are free of unfair bias toward certain sub-populations/groups	post	metamorphic testing	Black
ASTRAEA [190]	Debugging, Testing, Mitigation	how to perform fairness testing without an existing dataset, i.e., no training data access	post	grammar-based testing, metamorphic relations	Black
Aequitas [176]	Auditing, Analysis	how to audit for bias and fairness when developing and deploying algorithmic decision making systems	post	bias audit toolkit to support many bias metrics	Black
FairTest [202]	Testing, Debugging	how to detect unwarranted associations (UA) (disparate impact, offensive labels, and uneven error rates) between model outcomes and data attributes	post	unwarranted associations (UA) framework to determine UA between outcomes and attributes	Black
Themis-ml [19]	Fair Learning, Mitigation, Analysis, Auditing	how to measure, understand, and mitigate the implicit historical biases in socially sensitive data	pre, in & post	API for Fair ML for simple binary classifier	White, Black, & Grey
Fairify [29]	Verification Certification	how to verify individual fairness property in neural network (NN) models	in	SMT, formal analysis, pruning, input partitioning, interval arithmetic and activation heuristic	White

Generally, fairness analysis tools are mostly automated, they provide a software architecture, API or framework which implements several (mitigation) algorithms that enable developers to conduct bias analysis. For instance, AIF360 [22] provides an extensible architecture, FAT-Forensics [189] offers a python framework providing several mitigation algorithms, and Themis-ML [19] provides an API for analysis. Interestingly, some approaches allow to visualise fairness metrics while auditing or analyzing fairness properties (e.g., Fairkit-learn [112, 113]), and others enable fairness test experimentation. Notably, 2AFC [136] tests for (un)fairness via psycho-physical experimentation.

Fairness Analysis tools are mostly focused on the post validation of AI-based software, with little (grey) or no access (black) to the AI model. There is a need for tools that support white-box, in-processing stages of fairness analysis, especially for specification, formalization and verification.

Table 7. Details of Open Problems and Future Research Opportunities (more recent (2022 and 2023) solutions are in bracket)

Open Problems	Problem Description	Potential Solutions	Sample Related Work
Fairness Test Metrics and Adequacy	Measuring when fairness testing is sufficient/enough	Design of fairness test metrics and adequacy criteria	[61, 71, 123, 142, 162, 187] [144] ([146, 237])
Automatic Repair of Biased Classifiers	How to automatically repair biased classifiers to be (less or) un-biased?	Automatic Program Repair for fairness property	[9, 178, 195] ([104, 198])
Tooling for Fairness Property Specification	Specifying and engineering fairness properties for learning-based systems	Requirement Engineering tool support for Fairness properties	[21, 185] ([20, 72])
Unexplored or Poorly Understood Biases	Analyzing rare biases (e.g., age), complex or intersectional biases (e.g., age × gender)?	Fairness Analysis Support for rare, complex or intersectional Biases	[35, 38] ([49, 89])
Sequential and Long-term Fairness concerns	How to analyse/maintain fairness as the AI system evolves over time?	Techniques to support analysis of sequential and long-term fairness	[233] ([88, 164])
Human factors in fairness analysis	E.g., evaluating the harm induced by fairness violations to humans/society	Empirical studies of Human Factors in Fairness Analysis	[54, 100, 128] ([73, 77, 79, 92, 156])
Non-Specific/Holistic mitigation approaches	Designing bias mitigation methods that are agnostic of tasks, domains or datasets	General (i.e., task, domain and dataset -agnostic) bias analysis techniques	[232]
Fair Policy, Legalisation, and Compliance	How to design fairness analysis tools for policy makers and compliance officers?	Fairness Analysis Tool Support for Policy and Compliance Analysis	[136, 158]

6 LIMITATIONS AND THREATS TO VALIDITY

Collection, Filtering and Analysis of Publications: In this work, we have focused on in-depth analysis of publications exploring fairness as a software property or conducting fairness analysis via the lens of software engineering (SE). This scope means that we may have missed or filtered out papers that study fairness in other aspects, e.g., as a legal, ethical or transparency concern. Hence, this work is limited to the analysis of fairness property as an SE concern.

Manual Publication Analysis/Interpretation: The analysis of the publications studied in this paper are potentially open to human bias since they were manually coded and analyzed. However, to mitigate this threat we conduct both internal and external validation of the work. First, we ensure that the in-depth analysis and categorization of each paper is validated by at least one other researcher. In addition, we provide a copy of the paper and data to all (cited) authors for inspection and feedback. Feedback from cited authors improved our publication search and analysis. For instance, authors pointed out arxiv papers that have now been published (e.g., FairKit [112]), new publications missed, by our search due to string search (e.g., Thomas et al. [199]), and comprehensive or newer versions of cited papers (e.g., Fabris et al. [66]). Finally, we provide both the paper and data online to support scrutiny and reuse.

7 REFLECTION ON RECENT ADVANCES

In this section, we reflect on the recent advances in software fairness, since our study. We discuss the advancements made by recent works, how they relate to our previous findings and address previously open research challenges. We reflect on our initial findings with respect to recent publications in top-tier SE-specific literature, since our analysis. To this end, we highlight the following (13) recent (2024) publications, including papers from ICSE 2024 [49, 72, 131, 209], FSE 2024 [215], ISSTA 2024 [56, 217], TSE (2024) [222, 236], JSS 2024 [169] and TOSEM (2024) [46, 194, 194].

Intersectional Bias Testing : Our initial findings (**RQ2**) highlight the gap in fairness analysis of multiple attributes (aka intersectional bias) and unexplored complex biases (Table 7). In recent works, researchers have begun to tackle this problem. Notably, Chen et al. [49] presented an empirical study on fairness improvement for multiple protected attributes which shows that improving fairness for a single protected attribute can decrease fairness for other unconsidered protected attributes by up to 88.3%. This finding emphasizes the importance of studying intersectional fairness properties. Chen et al. [49] further conducted a large-scale benchmarking study to evaluate the effectiveness of state-of-the-art bias mitigation methods in improving intersectional fairness and assessing the trade-off between machine learning performance and intersectional fairness.

Fairness Requirements Engineering : Our work emphasized the low number of support for fairness requirements engineering (**RQ1**) and the lack of tooling for fairness specification (Table 7). However, some recent works have made significant effort to improve the state-of-the-art in this area. In 2024, Ferrara et al. [72] proposed a context-aware requirements engineering framework (called ReFair) that classifies sensitive features from user stories. ReFair employs NLP and word embedding analysis to recommend protected features to be considered during ML implementation.

Fairness-related Empirical Analysis : We had emphasized the low number of empirical analysis on software fairness metrics (**RQ1**), in particular, human studies (Table 7). We note that, in recent work, researchers have conducted empirical studies and human studies examining fairness concerns in bias mitigation and software engineering publication. Yang et al. [217] presents a recent empirical study comparing the performance of state-of-the-art fairness mitigation techniques for image classification. Using three datasets, five performance metrics and 13 mitigation methods, the authors demonstrate found that pre-processing methods and in-processing methods outperform post-processing methods, with pre-processing methods exhibiting the best performance. Liang et al. [135] investigates the effect of demographic information (gender and age) on the evaluation of technical articles on software engineering and potential behavioral differences among participants. The authors conducted a survey and human study, involving 540 participants, to investigate developers' evaluation of technical articles for software engineering. Results show that participants provide more positive content depth evaluations for younger male authors when compared to older male authors, male participants evaluate faster than female participants, and there is no significant difference in the genders of authors on the evaluation outcome of technical articles in SE.

Fairness Mitigation: In line with our previous observation (**RQ5**), bias mitigation methods remain a top focus in recent SE literature [56, 132, 215, 222]. Recently (2024), Xiao et al. [215] proposed MirrorFair an approach that employs a combination of model ensembling and counterfactual analysis to mitigate unfairness. MirrorFair constructs a counterfactual dataset from the original data and trains two models, one model on the original dataset and a second model on the counterfactual dataset. It then employs an ensemble of both model predictions to generate fairer decisions. Results show that MirrorFair outperforms baselines in fairness improvement, performance preservation, and trade-off metrics. Dasu et al. [56] also presented a fairness mitigation method (NeuFair) that employ a set of randomized algorithms that uses neuron dropout as a post-processing bias mitigation method. Results show that NeuFair is efficient and effective in improving fairness (up to 69%) with minimal performance degradation. Similarly, Li et al. [132] proposed model repair techniques that employ neural condition synthesis to repair biased systems. Yu et al. [222] proposed FairBalance, a pre-processing bias mitigation algorithm which balances the class distribution in each demographic group by assigning calculated weights to the training data. Fairbalance aims to identify the root cause of equalized odd violations and mitigate it without modifying the normal training process. Specifically, the authors observed that equalizing the class distribution in each demographic group with sample weights is a necessary condition for achieving equalized odds. Results show that FairBalance outperforms state-of-the-art approaches in terms of equalized odds.

Tabular and Semi-structured Datasets: Li et al. [131] addresses one of the open problems highlighted in our work (Table 7) – dataset-agnostic fairness analysis that cater for both tabular and semi-structured datasets. To achieve this, the authors propose a white-box fairness analysis approach called Responsible UNfair NEuron Repair (RUNNER). RUNNER improves the efficiency, effectiveness and generalization of existing works via Importance-based Neuron Diagnosis and Neuron Stabilizing Retraining. More importantly, RUNNER generalizes to both structured tabular data and large-scale unstructured image data, a combination which we observed as uncomon in previous works (**RQ4** and Table 7)

Fairness Testing: In our analysis, we emphasized previous works on fairness testing (**RQ1** and Table 6) and fairness metrics (e.g. fairness-performance trade-off (**RQ2**). There are some more recents works in both areas. In particular, Wang et al. [209] present a novel black-box individual fairness testing method called Model-Agnostic Fairness Testing (MAFT) which employs lightweight methods like gradient estimation and attribute perturbation for fairness testing. The authors show that MAFT achieves the same effectiveness as state-of-the-art white-box methods whilst improving the applicability to large-scale networks. Researchers have also proposed DistroFair to test class-level fairness in the presence of distributional shifts (i.e. ,out-of-distribution datasets) [169]. The authors concretize their approach for image datasets and demonstrate that it exposes class-level fairness in image datasets. Sun et al. [194] recently proposed FairMT an automated fairness testing approach for machine translation systems. FairMT aims to ensure that translations of semantically similar sentences containing protected attributes from distinct demographic groups maintain comparable meanings. The authors found that fair translations tend to exhibit superior translation performance, which challenges the existing literature on the existence of a tradeoff between fairness and performance.

Fairness Test Adequacy: We highlight the need for fairness test adequacy criteria (Table 7). A recent work studied this concern: Zheng et al. [236] conducted an empirical analysis to investigate the correlation between fairness property and test coverage criteria. Their study employed seven (7) coverage criteria, six (6) fairness metrics, three (3) testing techniques, five (5) bias mitigation methods, five (5) DNN models and nine (9) fairness datasets. Similar to our work, the authors found several open challenges in this area: They observed that (a) there is limited correlation between coverage criteria and fairness, (b) coverage and fairness metrics change as the test suite increases, and (c) models debiased by bias mitigation methods have a lower correlation, between coverage and fairness, compared to the original models.

Recent Closely-related Survey: In section 2, we compare our work to fairness-related surveys. However, more recently (2024), Chen et al. [46] presented a closely-related survey on fairness testing. This work is related to our work since fairness testing is one of the software development steps, and testing is inter-related to other software development steps, e.g., exposing fairness bugs is a check that fairness requirements are met. Unlike our work, Chen et al. [46] examine the literature with a focus on fairness testing workflow (how to test) and testing components (what to test). Similar to our work, Chen et al. [46] investigates the fairness-related literature, datasets, open source tools, research distributions, and research opportunities, but as it relates to fairness testing, instead of the entire SE pipeline (as done in this work). Notably, their work [46] presents several results that corroborate our findings. For instance, they also highlight the absence of works on intersectional bias (multiple sensitive attributes), the lack of test adequacy criteria for fairness testing and the lack of socio-technical and human-in-the-loop (stakeholder) support for fairness testing. Some of their findings are also complementary to ours. As an example, similar to our finding on unexplored or poorly understood biases (Table 7), Chen et al. [46] also highlight the difficulty of testing rare attributes and the challenge of fairness testing when sensitive attributes are absent (e.g., due to GDPR regulation). Unlike Chen et al. [46], we study software fairness beyond testing, we focus on its impact on the entire software development pipeline.

8 FUTURE OUTLOOK

This paper presents a comprehensive analysis and survey of publications on software fairness. In particular, we study publications that study *fairness as a software property*, works that examine fairness with a software engineering lens, or study how to engineer fair learning-based software systems. We identified several open problems and gaps. Table 7 highlights the details of the open problems identified in this work. Specifically, we highlight open problems in the

engineering of fair software systems, including concerns in the areas of fairness testing, verification and empirical evaluation of fairness properties. In the following we discuss the gaps and open problems we elicited from our analysis.

Even though there are several papers exploring fairness testing concerns [6, 203, 228] as well as works investigating test metrics for learning-based systems [105, 141, 193, 216], there is still the *problem of measuring test adequacy for fairness testing*. In our study, we found a few recent works [146, 237] *exploring the test adequacy of fairness testing approaches*: Zheng et al. [237] empirically studied the relationship between DNN fairness and neuron coverage test adequacy criteria. Their experiments showed that there is a limited statistical correlation between neuron overage criteria and DNN fairness, and these coverage criteria may be invalid for DNN fairness. Similarly, Majumder et al. [146] explored how to satisfy diverse fairness metrics and proposed a reduced set of fairness metrics. In particular, they cluster 30 metrics into seven clusters of classification metrics and three clusters of dataset metrics.

However, several questions concerning the adequacy of fairness testing remain unanswered and unexplored: When is fairness testing (in)sufficient? How can fairness testing be guided to reduce redundant test cases? Beyond fairness violations, are there other test metrics that indicate (in)sufficient testing has been performed? These are open problems in the area of fairness testing.

Despite the advances in test metrics for traditional and learning-based software, determining the appropriate test metrics or adequacy criteria for fairness evaluation remains an open problem. Researchers have proposed several test adequacy criteria for learning-based software, such as *neuron coverage* [162] and *surprise adequacy* [123]. *Neuron coverage* (from DeepXplore [162]) measures the parts of a learning-based systems that is exercised by a test input, it employs the ratio of neurons whose activation values were above a predefined threshold to measure the diversity of neuron behaviour and guide test generation. Likewise, *surprise adequacy* evaluates the behaviour of learning-based system with respect to their training data by measuring the surprise of an input as the difference in the system's behaviour between the input and the training data, such that a good test input should be sufficiently but not overtly surprising compared to the training data. Other test metrics for learning-based systems include DeepGini [71], Multiple-Boundary Clustering and Prioritization (MCP) [187] and Maximum Probability (MaxP) [144]. Despite the availability of several test metrics and adequacy criteria for functional testing of learning-based systems, there is no *indication or evaluation that demonstrates these test metrics are applicable for the fairness testing of learning-based systems*. The test metric typically employed by the fairness testing literature remains unfair behavior/outputs characterizing fairness violations. We encourage researchers to further investigate this vital challenge of fairness testing.

Although there are several fairness mitigation approaches, *repairing unfair learning-based systems to fulfill software fairness properties (i.e., to be fair or unbiased) has been hardly explored*, except for few works (e.g., Albarghouthi et al. [9], Li et al. [132], Tao et al. [198]). Albarghouthi et al. [9] proposed an approach that applies *distribution-guided inductive synthesis* to repair *unfair* ML classifiers with the goal of making them fair, it also verifies that the repaired classifiers is semantically close to the original (unfair) classifier. Their approach formulates the problem as a probability distribution problem, such that the repaired classifier needs to satisfy a probabilistic Boolean expression. Most importantly, this work is one of the few approaches we have found that aims to directly repair ML classifiers to fulfill fairness properties, without model re-training. Additionally, Tao et al. [198] have recently proposed model repair techniques that employ adversarial training to repair biased systems. Similarly, there is *low support for verification of fairness properties* (*see* RQ5). We had observed that verifying, certifying and providing guarantees for software fairness in learning-based software is under-explored.

Additional open problems include the *low number of empirical evaluation of fairness properties, especially as they relate to human factors and societal policies*. There are limited empirical studies studying or measuring the harm caused by

unfair software behavior (to humans) and the impact of fairness mitigation on marginalized individuals and communities. Blodgett et al. [32] emphasizes the importance of evaluating the harms induced by unfair (NLP) systems on humans.

Furthermore, there is *a need to develop approaches and tool support that are task, and dataset agnostic*. For instance, we observed that there is a lack of general, non-specific fairness mitigation approaches, most works target a specific domain/task (e.g., NLP, CV) with little work that is generally or demonstrably applicable across domains, tasks or datasets (except for few works like ADF [232]).

Besides, *there is a need to provide tools to support the automatic specification and requirements engineering of fairness properties*. Closely related works in this area includes MLCheck [185] and VeriFair [21] which allows to specify and check fairness specifications. However, these tools do not provide support for non-technical experts, e.g., compliance officers and policy makers. Indeed, such tools should support the definition and formalization of specific fairness properties as a socio-technical issue, not just a technical issue. These tools should allow not only allow to test fairness, but enable compliance officers and policy makers to audit and derive bias-preserving policies: For instance, equity-based policies have been proven to be useful in other domains, e.g., education (e.g., affirmative action) [158], toxic language detection [97]) and transport systems [174].

In RQ3, we demonstrate that there are some *poorly explored and understood biases, especially in terms of protected attributes*. Generally, we observed that certain tasks, datasets and biases are poorly studied. For instance, protected attributes relating to marital status, sexuality, non-binary gender, and education are poorly explored. Relating to this issue is the fact that the interactions of protected attributes is under-explored, especially in terms of the compounding or intersectional effect of multiple protected attributes, e.g., biases triggered by a combination of attributes (e.g., age × gender). While works like FairVis [38] allow to visualize and analyze intersectional bias, there is little support for specifying, testing and mitigating such biases.

Sequential and long-term fairness concerns also remain an open problem [233]. For instance, how do we mitigate fairness properties as the software system evolves? Do the proposed mitigation approaches for one-time fairness analysis scale to sequential or long-term concerns.

In addition, there is the socio-technical and ethical concern of fairness analysis: How do we ensure that our fairness mitigation approaches do not induce new and unintended biases? How do our proposed mitigation and analysis approaches translate to real-world intervention for fairness, e.g., in terms of equity-based mitigation approaches typically employed in public policy (e.g., affirmative action [158])? In summary, we posit that there is a need for socio-technical, human-in-the-loop bias analysis approaches that translate to the mitigation of real-world harms to humans and society [77]. There is a need to provide bias analysis methods to support developers, policy makers and compliance officers.

9 CONCLUSION

This paper presents a survey of the literature on software fairness analysis. It is particularly focused on examining the landscape of the research works that explore fairness with the lens of software engineering. We initially collected 164 papers in our initial analysis (2010-2021) and examine several aspects of the literature including the goals of the available work. This includes the focus of the literature in terms of the domains, measures, attributes and datasets that are explored. Additionally, we examine the under-examined or unexplored areas and discuss open research opportunities in software fairness. We also collected an additional 41 papers (2022-2023), from the dominant (SE) venues. We discuss the most recent advancements in software fairness analysis since our initial study. For easy scrutiny, reuse and replication, we provide details of collected papers, our in-depth analysis for each research question and findings:

https://github.com/ezekiel-soremekun/Software-Fairness-Analysis

ACKNOWLEDGMENTS

We would like to thank the reviewers for their valuable feedback, which has improved our analysis and paper. In addition, we shared our work with all authors of the (cited) papers, we would like to thank the authors who provided feedback on previous drafts of this paper. This research was funded by the Luxembourg National Research Fund (FNR), grant reference NCER22/IS/16570468/NCER-FT.

REFERENCES

[1] Abubakar Abid, Maheen Farooqi, and James Zou. 2021. Persistent anti-muslim bias in large language models. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*. 298–306.

[2] Serge Abiteboul and Julia Stoyanovich. 2019. Transparency, fairness, data protection, neutrality: Data management challenges in the face of new regulation. *Journal of Data and Information Quality (JDIQ)* 11, 3 (2019), 1–9.

[3] Julius A Adebayo et al. 2016. *FairML: ToolBox for diagnosing bias in predictive modeling*. Ph.D. Dissertation. Massachusetts Institute of Technology.

[4] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. 2018. A reductions approach to fair classification. In *International Conference on Machine Learning*. PMLR, 60–69.

[5] Chirag Agarwal, Himabindu Lakkaraju, and Marinka Zitnik. 2021. Towards a unified framework for fair and stable graph representation learning. In *Uncertainty in Artificial Intelligence*. PMLR, 2114–2124.

[6] Aniya Aggarwal, Pranay Lohia, Seema Nagar, Kuntal Dey, and Diptikalyan Saha. 2019. Black box fairness testing of machine learning models. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 625–635.

[7] Aniya Aggarwal, Samiulla Shaikh, Sandeep Hans, Swastik Haldar, Rema Ananthanarayanan, and Diptikalyan Saha. 2021. Testing framework for black-box AI models. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. IEEE, 81–84.

[8] Aws Albarghouthi, Loris D'Antoni, Samuel Drews, and Aditya V Nori. 2017. Fairsquare: probabilistic verification of program fairness. *Proceedings of the ACM on Programming Languages* 1, OOPSLA (2017), 1–30.

[9] Aws Albarghouthi, Loris D'Antoni, and Samuel Drews. 2017. Repairing decision-making programs under uncertainty. In *International Conference on Computer Aided Verification*. Springer, 181–200.

[10] Aws Albarghouthi and Samuel Vinitsky. 2019. Fairness-aware programming. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. 211–219.

[11] Jose Manuel Alvarez and Salvatore Ruggieri. 2023. Counterfactual Situation Testing: Uncovering Discrimination under Fairness given the Difference. In *Proceedings of the 3rd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '23)*. Association for Computing Machinery, New York, NY, USA, Article 2, 11 pages. https://doi.org/10.1145/3617694.3623222

[12] Rico Angell, Brittany Johnson, Yuriy Brun, and Alexandra Meliou. 2018. Themis: Automatically testing software for discrimination. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 871–875.

[13] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. 2022. Machine bias. In *Ethics of data and analytics*. Auerbach Publications, 254–264.

[14] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Accessed:09.11.2023. Machine Bias: There's software used across the country to predict future criminals. And it's biased against blacks. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

[15] Carolyn Ashurst and Adrian Weller. 2023. Fairness Without Demographic Data: A Survey of Approaches. In *Proceedings of the 3rd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization*. 1–12.

[16] Carolyn Ashurst and Adrian Weller. 2023. Fairness Without Demographic Data: A Survey of Approaches. In *Proceedings of the 3rd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '23)*. Association for Computing Machinery, New York, NY, USA, Article 14, 12 pages. https://doi.org/10.1145/3617694.3623234

[17] Muhammad Hilmi Asyrofi, Zhou Yang, Imam Nur Bani Yusuf, Hong Jin Kang, Ferdian Thung, and David Lo. 2021. Biasfinder: Metamorphic test generation to uncover bias for sentiment analysis systems. *IEEE Transactions on Software Engineering* (2021).

[18] Fatma Basak Aydemir and Fabiano Dalpiaz. 2018. A roadmap for ethics-aware software engineering. In *2018 IEEE/ACM International Workshop on Software Fairness (FairWare)*. IEEE, 15–21.

[19] Niels Bantilan. 2018. Themis-ml: A fairness-aware machine learning interface for end-to-end discrimination discovery and mitigation. *Journal of Technology in Human Services* 36, 1 (2018), 15–30.

[20] Luciano Baresi, Chiara Criscuolo, and Carlo Ghezzi. 2023. Understanding fairness requirements for ml-based software. In *2023 IEEE 31st International Requirements Engineering Conference (RE)*. IEEE, 341–346.

[21] Osbert Bastani, Xin Zhang, and Armando Solar-Lezama. 2019. Probabilistic verification of fairness properties via concentration. *Proceedings of the ACM on Programming Languages* 3, OOPSLA (2019), 1–27.

[22] Rachel KE Bellamy, Kuntal Dey, Michael Hind, Samuel C Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilović, et al. 2019. AI Fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM Journal of Research and Development* 63, 4/5 (2019), 4–1.

[23] Rachel KE Bellamy, Kuntal Dey, Michael Hind, Samuel C Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, et al. 2019. Think your artificial intelligence software is fair? Think again. *IEEE Software* 36, 4 (2019), 76–80.

[24] Nelly Bencomo, Jin LC Guo, Rachel Harrison, Hans-Martin Heyn, and Tim Menzies. 2021. The Secret to Better AI and Better Software (Is Requirements Engineering). *IEEE Software* 39, 1 (2021), 105–110.

[25] Richard Berk, Hoda Heidari, Shahin Jabbari, Matthew Joseph, Michael Kearns, Jamie Morgenstern, Seth Neel, and Aaron Roth. 2017. A Convex Framework for Fair Regression. *Fairness, Accountability, and Transparency in Machine Learning* (2017).

[26] Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. 2020. Fairlearn: A toolkit for assessing and improving fairness in AI. *Microsoft, Tech. Rep. MSR-TR-2020-32* (2020).

[27] Sumon Biswas and Hridesh Rajan. 2020. Do the machine learning models on a crowd sourced platform exhibit bias? an empirical study on model fairness. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 642–653.

[28] Sumon Biswas and Hridesh Rajan. 2021. Fair Preprocessing: Towards Understanding Compositional Fairness of Data Transformers in Machine Learning Pipeline. *arXiv preprint arXiv:2106.06054* (2021).

[29] Sumon Biswas and Hridesh Rajan. 2023. Fairify: Fairness verification of neural networks. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 1546–1558.

[30] Emily Black, Rakshit Naidu, Rayid Ghani, Kit Rodolfa, Daniel Ho, and Hoda Heidari. 2023. Toward Operationalizing Pipeline-aware ML Fairness: A Research Agenda for Developing Practical Guidelines and Tools. In *Proceedings of the 3rd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '23)*. Association for Computing Machinery, New York, NY, USA, Article 36, 11 pages. https://doi.org/10.1145/3617694.3623259

[31] Emily Black, Samuel Yeom, and Matt Fredrikson. 2020. Fliptest: fairness testing via optimal transport. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 111–121.

[32] Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (Technology) is Power: A Critical Survey of "Bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 5454–5476.

[33] Su Lin Blodgett, Gilsinia Lopez, Alexandra Olteanu, Robert Sim, and Hanna Wallach. 2021. Stereotyping Norwegian salmon: an inventory of pitfalls in fairness benchmark datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 1004–1015.

[34] C. Malik Boykin, Sophia T. Dasch, Vincent Rice Jr., Venkat R. Lakshminarayanan, Taiwo A. Togun, and Sarah M. Brown. 2021. Opportunities for a More Interdisciplinary Approach to Measuring Perceptions of Fairness in Machine Learning. In *Proceedings of the 1st ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization* (–, NY, USA) *(EAAMO '21)*. Association for Computing Machinery, New York, NY, USA, Article 1, 9 pages. https://doi.org/10.1145/3465416.3483302

[35] Martim Brandao. 2019. Age and gender bias in pedestrian detection algorithms. In *Proceedings of the Workshop on Fairness Accountability Transparency and Ethics in Computer Vision at IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*.

[36] Yuriy Brun and Alexandra Meliou. 2018. Software fairness. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 754–759.

[37] Margaret Burnett, Simone Stumpf, Jamie Macbeth, Stephann Makri, Laura Beckwith, Irwin Kwan, Anicia Peters, and William Jernigan. 2016. GenderMag: A method for evaluating software's gender inclusiveness. *Interacting with Computers* 28, 6 (2016), 760–787.

[38] Ángel Alexander Cabrera, Will Epperson, Fred Hohman, Minsuk Kahng, Jamie Morgenstern, and Duen Horng Chau. 2019. FairVis: Visual analytics for discovering intersectional bias in machine learning. In *2019 IEEE Conference on Visual Analytics Science and Technology (VAST)*. IEEE, 46–56.

[39] Toon Calders and Sicco Verwer. 2010. Three naive Bayes approaches for discrimination-free classification. *Data mining and knowledge discovery* 21, 2 (2010), 277–292.

[40] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. 2017. Optimized pre-processing for discrimination prevention. *Advances in neural information processing systems* 30 (2017).

[41] Simon Caton and Christian Haas. 2020. Fairness in machine learning: A survey. *arXiv preprint arXiv:2010.04053* (2020).

[42] L Elisa Celis, Lingxiao Huang, Vijay Keswani, and Nisheeth K Vishnoi. 2019. Classification with fairness constraints: A meta-algorithm with provable guarantees. In *Proceedings of the conference on fairness, accountability, and transparency*. 319–328.

[43] Joymallya Chakraborty, Suvodeep Majumder, and Tim Menzies. 2021. Bias in machine learning software: why? how? what to do?. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 429–440.

[44] Joymallya Chakraborty, Suvodeep Majumder, Zhe Yu, and Tim Menzies. 2020. Fairway: A way to build fair ml software. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 654–665.

[45] Jialuo Chen, Jingyi Wang, Xingjun Ma, Youcheng Sun, Jun Sun, Peixin Zhang, and Peng Cheng. 2023. QuoTe: Quality-Oriented Testing for Deep Learning Systems. *ACM Trans. Softw. Eng. Methodol.* 32, 5, Article 125 (jul 2023), 33 pages. https://doi.org/10.1145/3582573

[46] Zhenpeng Chen, Jie M Zhang, Max Hort, Mark Harman, and Federica Sarro. 2024. Fairness testing: A comprehensive survey and analysis of trends. *ACM Transactions on Software Engineering and Methodology* 33, 5 (2024), 1–59.

[47] Zhenpeng Chen, Jie M Zhang, Federica Sarro, and Mark Harman. 2022. MAAT: a novel ensemble approach to addressing fairness and performance bugs for machine learning software. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 1122–1134.

[48] Zhenpeng Chen, Jie M. Zhang, Federica Sarro, and Mark Harman. 2023. A Comprehensive Empirical Study of Bias Mitigation Methods for Machine Learning Classifiers. *ACM Trans. Softw. Eng. Methodol.* 32, 4, Article 106 (may 2023), 30 pages. https://doi.org/10.1145/3583561

[49] Zhenpeng Chen, Jie M Zhang, Federica Sarro, and Mark Harman. 2024. Fairness improvement with multiple protected attributes: How far are we? (2024), 1–13.

[50] Lu Cheng, Kush R Varshney, and Huan Liu. 2021. Socially responsible ai algorithms: Issues, purposes, and challenges. *Journal of Artificial Intelligence Research* 71 (2021), 1137–1181.

[51] Jürgen Cito, Isil Dillig, Seohyun Kim, Vijayaraghavan Murali, and Satish Chandra. 2021. Explaining mispredictions of machine learning models using rule induction. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 716–727.

[52] Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. 2017. Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd acm sigkdd international conference on knowledge discovery and data mining*. 797–806.

[53] Kate Crawford. 2017. The trouble with bias. In *Conference on Neural Information Processing Systems, Invited Speaker*. https://www.youtube.com/watch?v=fMym_BKWQzk

[54] Jenna Cryan, Shiliang Tang, Xinyi Zhang, Miriam Metzger, Haitao Zheng, and Ben Y Zhao. 2020. Detecting gender stereotypes: lexicon vs. supervised learning methods. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–11.

[55] Anubrata Das and Matthew Lease. 2019. A Conceptual Framework for Evaluating Fairness in Search. *arXiv preprint arXiv:1907.09328* (2019).

[56] Vishnu Asutosh Dasu, Ashish Kumar, Saeid Tizpaz-Niari, and Gang Tan. 2024. NeuFair: Neural Network Fairness Repair with Dropout. In *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*. 1541–1553.

[57] Terrance De Vries, Ishan Misra, Changhan Wang, and Laurens Van der Maaten. 2019. Does object recognition work for everyone?. In *Proceedings of the Workshop on Fairness Accountability Transparency and Ethics in Computer Vision at IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 52–59.

[58] Isin Demirsahin, Oddur Kjartansson, Alexander Gutkin, and Clara Rivera. 2020. Open-source multi-speaker corpora of the english accents in the british isles. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*. 6532–6541.

[59] Emily Denton, Ben Hutchinson, Margaret Mitchell, and Timnit Gebru. 2019. Detecting bias with generative counterfactual face attribute augmentation. (2019).

[60] Marina Drosou, HV Jagadish, Evaggelia Pitoura, and Julia Stoyanovich. 2017. Diversity in big data: A review. *Big data* 5, 2 (2017), 73–84.

[61] Xiaoning Du, Xiaofei Xie, Yi Li, Lei Ma, Yang Liu, and Jianjun Zhao. 2019. DeepStellar: Model-Based Quantitative Analysis of Stateful Deep Learning Systems. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Tallinn, Estonia) *(ESEC/FSE 2019)*. Association for Computing Machinery, New York, NY, USA, 477–487. https://doi.org/10.1145/3338906.3338954

[62] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*. 214–226.

[63] Michael D Ekstrand, Robin Burke, and Fernando Diaz. 2019. Fairness and discrimination in recommendation and retrieval. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 576–577.

[64] Simone Fabbrizzi, Symeon Papadopoulos, Eirini Ntoutsi, and Ioannis Kompatsiaris. 2021. A Survey on Bias in Visual Datasets. *arXiv preprint arXiv:2107.07919* (2021).

[65] Alessandro Fabris, Fabio Giachelle, Alberto Piva, Gianmaria Silvello, and Gian Antonio Susto. 2023. A Search Engine for Algorithmic Fairness Datasets. In *Proceedings of the 2nd European Workshop on Algorithmic Fairness, Winterthur, Switzerland, June 7th to 9th, 2023 (CEUR Workshop Proceedings, Vol. 3442)*, Jose M. Alvarez, Alessandro Fabris, Christoph Heitz, Corinna Hertweck, Michele Loi, and Meike Zehlike (Eds.). CEUR-WS.org. https://ceur-ws.org/Vol-3442/paper-08.pdf

[66] Alessandro Fabris, Stefano Messina, Gianmaria Silvello, and Gian Antonio Susto. 2022. Algorithmic fairness datasets: the story so far. *Data Min. Knowl. Discov.* 36, 6 (2022), 2074–2152. https://doi.org/10.1007/S10618-022-00854-Z

[67] Alessandro Fabris, Stefano Messina, Gianmaria Silvello, and Gian Antonio Susto. 2022. Tackling Documentation Debt: A Survey on Algorithmic Fairness Datasets. In *Proceedings of the 2nd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '22)*. Association for Computing Machinery, New York, NY, USA, Article 2, 13 pages. https://doi.org/10.1145/3551624.3555286

[68] Ming Fan, Wenying Wei, Wuxia Jin, Zijiang Yang, and Ting Liu. 2022. Explanation-Guided Fairness Testing through Genetic Algorithm. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. IEEE.

[69] Golnoosh Farnad, Behrouz Babaki, and Michel Gendreau. 2020. A unifying framework for fairness-aware influence maximization. In *Companion Proceedings of the Web Conference 2020*. 714–722.

[70] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 259–268.

[71] 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

[72] Carmine Ferrara, Francesco Casillo, Carmine Gravino, Andrea De Lucia, and Fabio Palomba. 2024. Refair: Toward a context-aware recommender for fairness requirements engineering. (2024), 1–12.

[73] Carmine Ferrara, Giulia Sellitto, Filomena Ferrucci, Fabio Palomba, and Andrea De Lucia. 2023. Fairness-aware machine learning engineering: how far are we? *Empirical Software Engineering* 29, 1 (2023), 9.

[74] Anjalie Field, Su Lin Blodgett, Zeerak Waseem, and Yulia Tsvetkov. 2021. A Survey of Race, Racism, and Anti-Racism in NLP. *arXiv preprint arXiv:2106.11410* (2021).

[75] Anthony Finkelstein, Mark Harman, S Afshin Mansouri, Jian Ren, and Yuanyuan Zhang. 2009. A search based approach to fairness analysis in requirement assignments to aid negotiation, mediation and decision making. *Requirements engineering* 14, 4 (2009), 231–245.

[76] Jessie Finocchiaro, Roland Maio, Faidra Monachou, Gourab K Patro, Manish Raghavan, Ana-Andreea Stoica, and Stratis Tsirtsis. 2021. Bridging machine learning and mechanism design towards algorithmic fairness. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. 489–503.

[77] Claudia Flores-Saviaga, Christopher Curtis, and Saiph Savage. 2023. Inclusive Portraits: Race-Aware Human-in-the-Loop Technology. In *Proceedings of the 3rd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '23)*. Association for Computing Machinery, New York, NY, USA, Article 15, 11 pages. https://doi.org/10.1145/3617694.3623235

[78] Batya Friedman and Helen Nissenbaum. 1996. Bias in computer systems. *ACM Transactions on Information Systems (TOIS)* 14, 3 (1996), 330–347.

[79] Aimen Gaba, Zhanna Kaufman, Jason Cheung, Marie Shvakel, Kyle Wm Hall, Yuriy Brun, and Cindy Xiong Bearfield. 2023. My Model is Unfair, Do People Even Care? Visual Design Affects Trust and Perceived Bias in Machine Learning. *IEEE Transactions on Visualization and Computer Graphics* (2023).

[80] Pratik Gajane and Mykola Pechenizkiy. 2017. On formalizing fairness in prediction with machine learning. *arXiv preprint arXiv:1710.03184* (2017).

[81] Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou. 2017. Fairness testing: testing software for discrimination. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. 498–510.

[82] Xuanqi Gao, Juan Zhai, Shiqing Ma, Chao Shen, Yufei Chen, and Qian Wang. 2022. FairNeuron: Improving Deep Neural Network Fairness with Adversary Games on Selective Neurons. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. IEEE.

[83] Xuanqi Gao, Juan Zhai, Shiqing Ma, Chao Shen, Yufei Chen, and Shiwei Wang. 2023. CILIATE: Towards Fairer Class-based Incremental Learning by Dataset and Training Refinement. *arXiv preprint arXiv:2304.04222* (2023).

[84] Tanmay Garg, Sarah Masud, Tharun Suresh, and Tanmoy Chakraborty. 2023. Handling bias in toxic speech detection: A survey. *Comput. Surveys* 55, 13s (2023), 1–32.

[85] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. 2021. Datasheets for datasets. *Commun. ACM* 64, 12 (2021), 86–92.

[86] Bishwamittra Ghosh, Debabrota Basu, and Kuldeep S Meel. 2021. Justicia: A Stochastic SAT Approach to Formally Verify Fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 7554–7563.

[87] Stephen Giguere, Blossom Metevier, Yuriy Brun, Bruno Castro Da Silva, Philip S Thomas, and Scott Niekum. 2022. Fairness guarantees under demographic shift. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*.

[88] Usman Gohar, Sumon Biswas, and Hridesh Rajan. 2023. Towards understanding fairness and its composition in ensemble machine learning. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 1533–1545.

[89] Usman Gohar and Lu Cheng. 2023. A Survey on Intersectional Fairness in Machine Learning: Notions, Mitigation, and Challenges. *arXiv preprint arXiv:2305.06969* (2023).

[90] CV González Zelaya, P Missier, and D Prangle. 2019. Parametrised data sampling for fairness optimisation. In *2019 XAI Workshop at SIGKDD, Anchorage, AK, USA*.

[91] Nina Grgic-Hlaca, Elissa M Redmiles, Krishna P Gummadi, and Adrian Weller. 2018. Human perceptions of fairness in algorithmic decision making: A case study of criminal risk prediction. In *Proceedings of the 2018 world wide web conference*. 903–912.

[92] Nina Grgić-Hlača, Gabriel Lima, Adrian Weller, and Elissa M. Redmiles. 2022. Dimensions of Diversity in Human Perceptions of Algorithmic Fairness. In *Proceedings of the 2nd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '22)*. Association for Computing Machinery, New York, NY, USA, Article 21, 12 pages. https://doi.org/10.1145/3551624.3555306

[93] Huizhong Guo. 2023. Fairness testing for recommender systems. In *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*. 1546–1548.

[94] Huizhong Guo, Jinfeng Li, Jingyi Wang, Xiangyu Liu, Dongxia Wang, Zehong Hu, Rong Zhang, and Hui Xue. 2023. FairRec: Fairness Testing for Deep Recommender Systems. *arXiv preprint arXiv:2304.07030* (2023).

[95] Wenshuo Guo, Karl Krauth, Michael Jordan, and Nikhil Garg. 2021. The Stereotyping Problem in Collaboratively Filtered Recommender Systems. In *Proceedings of the 1st ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization* (–, NY, USA) *(EAAMO '21)*. Association for Computing Machinery, New York, NY, USA, Article 6, 10 pages. https://doi.org/10.1145/3465416.3483298

[96] Emitzá Guzmán, Ricarda Anna-Lena Fischer, and Janey Kok. 2023. Mind the gap: gender, micro-inequities and barriers in software development. *Empirical Software Engineering* 29, 1 (2023), 17.

[97] Matan Halevy, Camille Harris, Amy Bruckman, Diyi Yang, and Ayanna Howard. 2021. Mitigating Racial Biases in Toxic Language Detection with an Equity-Based Ensemble Framework. In *Proceedings of the 1st ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization* (–,

NY, USA) *(EAAMO '21)*. Association for Computing Machinery, New York, NY, USA, Article 7, 11 pages. https://doi.org/10.1145/3465416.3483299

[98] Galen Harrison, Julia Hanson, Christine Jacinto, Julio Ramirez, and Blase Ur. 2020. An empirical study on the perceived fairness of realistic, imperfect machine learning models. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*. 392–402.

[99] Austin Hoag, James E. Kostas, Bruno Castro da Silva, Philip S. Thomas, and Yuriy Brun. 2023. Seldonian Toolkit: Building Software with Safe and Fair Machine Learning. In *2023 IEEE/ACM 45th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. 107–111. https://doi.org/10.1109/ICSE-Companion58688.2023.00035

[100] Kenneth Holstein, Jennifer Wortman Vaughan, Hal Daumé III, Miro Dudik, and Hanna Wallach. 2019. Improving fairness in machine learning systems: What do industry practitioners need?. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. 1–16.

[101] Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. 2020. Characterising bias in compressed models. *arXiv preprint arXiv:2010.03058* (2020).

[102] Max Hort, Rebecca Moussa, and Federica Sarro. 2023. Multi-objective search for gender-fair and semantically correct word embeddings. *Applied Soft Computing* 133 (2023), 109916.

[103] Max Hort, Jie M Zhang, Federica Sarro, and Mark Harman. 2021. Fairea: a model behaviour mutation approach to benchmarking bias mitigation methods. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 994–1006.

[104] Max Hort, Jie M. Zhang, Federica Sarro, and Mark Harman. 2024. Search-based Automatic Repair for Fairness and Accuracy in Decision-making Software. *Empirical Software Engineering* 29, 1 (2024), 36. https://doi.org/10.1007/s10664-023-10419-3

[105] Chao Huang, Junbo Zhang, Yu Zheng, and Nitesh V. Chawla. 2018. DeepCrime: Attentive Hierarchical Recurrent Networks for Crime Prediction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (Torino, Italy) *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 1423–1432. https://doi.org/10.1145/3269206.3271793

[106] Dong Huang, Qingwen Bu, Jie Zhang, Xiaofei Xie, Junjie Chen, and Heming Cui. 2023. Bias assessment and mitigation in llm-based code generation. *arXiv preprint arXiv:2309.14345* (2023).

[107] Waqar Hussain, Davoud Mougouei, and Jon Whittle. 2018. Integrating social values into software design patterns. In *2018 IEEE/ACM International Workshop on Software Fairness (FairWare)*. IEEE, 8–14.

[108] Ben Hutchinson and Margaret Mitchell. 2019. 50 years of test (un) fairness: Lessons for machine learning. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. 49–58.

[109] Wiebke (Toussaint) Hutiri, Aaron Yi Ding, Fahim Kawsar, and Akhil Mathur. 2023. Tiny, Always-on, and Fragile: Bias Propagation through Design Choices in On-Device Machine Learning Workflows. *ACM Trans. Softw. Eng. Methodol.* 32, 6, Article 155 (sep 2023), 37 pages. https://doi.org/10.1145/3591867

[110] HV Jagadish, Julia Stoyanovich, and Bill Howe. 2021. Covid-19 brings data equity challenges to the fore. *Digital Government: Research and Practice* 2, 2 (2021), 1–7.

[111] Philips George John, Deepak Vijaykeerthy, and Diptikalyan Saha. 2020. Verifying individual fairness in machine learning models. In *Conference on Uncertainty in Artificial Intelligence*. PMLR, 749–758.

[112] Brittany Johnson, Jesse Bartola, Rico Angell, Sam Witty, Stephen Giguere, and Yuriy Brun. 2023. Fairkit, fairkit, on the wall, who's the fairest of them all? Supporting fairness-related decision-making. *EURO Journal on Decision Processes* 11 (2023), 100031. https://doi.org/10.1016/j.ejdp.2023.100031

[113] Brittany Johnson and Yuriy Brun. 2022. Fairkit-Learn: A Fairness Evaluation and Comparison Toolkit. In *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings* (Pittsburgh, Pennsylvania) *(ICSE '22)*. Association for Computing Machinery, New York, NY, USA, 70–74. https://doi.org/10.1145/3510454.3516830

[114] Faisal Kamiran and Toon Calders. 2009. Classifying without discriminating. In *2009 2nd international conference on computer, control and communication*. IEEE, 1–6.

[115] Faisal Kamiran and Toon Calders. 2012. Data preprocessing techniques for classification without discrimination. *Knowledge and information systems* 33, 1 (2012), 1–33.

[116] Faisal Kamiran, Toon Calders, and Mykola Pechenizkiy. 2010. Discrimination aware decision tree learning. In *2010 IEEE International Conference on Data Mining*. IEEE, 869–874.

[117] Faisal Kamiran, Asim Karim, and Xiangliang Zhang. 2012. Decision theory for discrimination-aware classification. In *2012 IEEE 12th International Conference on Data Mining*. IEEE, 924–929.

[118] Faisal Kamiran, Sameen Mansha, Asim Karim, and Xiangliang Zhang. 2018. Exploiting reject option in classification for social discrimination control. *Information Sciences* 425 (2018), 18–33.

[119] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2012. Fairness-aware classifier with prejudice remover regularizer. In *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 35–50.

[120] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. 2018. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*. PMLR, 2564–2572.

[121] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. 2019. An empirical study of rich subgroup fairness for machine learning. In *Proceedings of the conference on fairness, accountability, and transparency*. 100–109.

[122] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. 2019. Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9012–9020.

[123] 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, 1039–1049.

[124] Michael P Kim, Amirata Ghorbani, and James Zou. 2019. Multiaccuracy: Black-box post-processing for fairness in classification. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*. 247–254.

[125] Barbara Kitchenham. 2004. Procedures for performing systematic reviews. *Keele, UK, Keele University* 33, 2004 (2004), 1–26.

[126] Caitlin Kuhlman, MaryAnn VanValkenburg, and Elke Rundensteiner. 2019. Fare: Diagnostics for fair ranking using pairwise error metrics. In *The World Wide Web Conference*. 2936–2942.

[127] Marta Z Kwiatkowska. 1989. Survey of fairness notions. *Information and Software Technology* 31, 7 (1989), 371–386.

[128] Michelle Seng Ah Lee and Jat Singh. 2021. The landscape and gaps in open source fairness toolkits. In *Proceedings of the 2021 CHI conference on human factors in computing systems*. 1–13.

[129] Grace A Lewis, Ipek Ozkaya, and Xiwei Xu. 2021. Software architecture challenges for ml systems. In *2021 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 634–638.

[130] Chenglu Li, Wanli Xing, and Walter Leite. 2021. Yet Another Predictive Model? Fair Predictions of Students' Learning Outcomes in an Online Math Learning Platform. In *LAK21: 11th International Learning Analytics and Knowledge Conference*. 572–578.

[131] Tianlin Li, Yue Cao, Jian Zhang, Shiqian Zhao, Yihao Huang, Aishan Liu, Qing Guo, and Yang Liu. 2024. Runner: Responsible unfair neuron repair for enhancing deep neural network fairness. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*. 1–13.

[132] Tianlin Li, Xiaofei Xie, Jian Wang, Qing Guo, Aishan Liu, Lei Ma, and Yang Liu. 2023. Faire: Repairing Fairness of Neural Networks via Neuron Condition Synthesis. *ACM Trans. Softw. Eng. Methodol.* 33, 1, Article 21 (nov 2023), 24 pages. https://doi.org/10.1145/3617168

[133] Xinyue Li, Zhenpeng Chen, Jie M Zhang, Federica Sarro, Ying Zhang, and Xuanzhe Liu. 2023. Dark-skin individuals are at more risk on the street: Unmasking fairness issues of autonomous driving systems. *arXiv preprint arXiv:2308.02935* (2023).

[134] Yanhui Li, Linghan Meng, Lin Chen, Li Yu, Di Wu, Yuming Zhou, and Baowen Xu. 2022. Training Data Debugging for the Fairness of Machine Learning Software. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. IEEE.

[135] Anda Liang, Emerson Murphy-Hill, Westley Weimer, and Yu Huang. 2024. A Controlled Experiment in Age and Gender Bias When Reading Technical Articles in Software Engineering. *IEEE Trans. Softw. Eng.* 50, 10 (Oct. 2024), 2498–2511. https://doi.org/10.1109/TSE.2024.3437355

[136] Lizhen Liang and Daniel E Acuna. 2020. Artificial mental phenomena: Psychophysics as a framework to detect perception biases in AI models. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*. 403–412.

[137] Zachary Lipton, Julian McAuley, and Alexandra Chouldechova. 2018. Does mitigating ML's impact disparity require treatment disparity? *Advances in neural information processing systems* 31 (2018).

[138] Ye Liu, Yi Li, Shang-Wei Lin, and Rong Zhao. 2020. Towards automated verification of smart contract fairness. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 666–677.

[139] Steven R Livingstone and Frank A Russo. 2018. The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. *PloS one* 13, 5 (2018), e0196391.

[140] Kristian Lum and Tarak Shah. 2019. Measures of fairness for New York City's Supervised Release Risk Assessment Tool. *Human Rights Data Analytics Group* (2019), 21.

[141] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. 2018. DeepGauge: Multi-Granularity Testing Criteria for Deep Learning Systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering* (Montpellier, France) *(ASE 2018)*. Association for Computing Machinery, New York, NY, USA, 120–131. https://doi.org/10.1145/3238147.3238202

[142] Lei Ma, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Felix Juefei-Xu, Chao Xie, Li Li, Yang Liu, Jianjun Zhao, et al. 2018. Deepmutation: Mutation testing of deep learning systems. In *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 100–111.

[143] Pingchuan Ma, Shuai Wang, and Jin Liu. 2020. Metamorphic Testing and Certified Mitigation of Fairness Violations in NLP Models.. In *IJCAI*. 458–465.

[144] Wei Ma, Mike Papadakis, Anestis Tsakmalis, Maxime Cordy, and Yves Le Traon. 2021. Test Selection for Deep Learning Systems. *ACM Trans. Softw. Eng. Methodol.* 30, 2, Article 13 (Jan. 2021), 22 pages. https://doi.org/10.1145/3417330

[145] Mohammad Mahdi Mohajer, Alvine Boaye Belle, Junjie Wang, Hadi Hemmati, Song Wang, Zhen Ming, et al. 2023. A First Look at Fairness of Machine Learning Based Code Reviewer Recommendation. *arXiv e-prints* (2023), arXiv–2307.

[146] Suvodeep Majumder, Joymallya Chakraborty, Gina R. Bai, Kathryn T. Stolee, and Tim Menzies. 2023. Fair Enough: Searching for Sufficient Measures of Fairness. *ACM Trans. Softw. Eng. Methodol.* 32, 6, Article 134 (sep 2023), 22 pages. https://doi.org/10.1145/3585006

[147] Karima Makhlouf, Sami Zhioua, and Catuscia Palamidessi. 2020. Survey on Causal-based Machine Learning Fairness Notions. *arXiv preprint arXiv:2010.09553* (2020).

[148] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)* 54, 6 (2021), 1–35.

[149] Blossom Metevier, Stephen Giguere, Sarah Brockman, Ari Kobren, Yuriy Brun, Emma Brunskill, and Philip Thomas. 2019. Offline contextual bandits with high probability fairness guarantees. *Advances in neural information processing systems* 32 (2019).

[150] Verya Monjezi, Ashutosh Trivedi, Gang Tan, and Saeid Tizpaz-Niari. 2023. Information-Theoretic Testing and Debugging of Fairness Defects in Deep Neural Networks. *arXiv preprint arXiv:2304.04199* (2023).

[151] Daniel Perez Morales, Takashi Kitamura, and Shingo Takada. 2021. Coverage-Guided Fairness Testing. In *International Conference on Intelligence Science*. Springer, 183–199.

[152] Henry Muccini and Karthik Vaidhyanathan. 2021. Software architecture for ML-based systems: What exists and what lies ahead. In *2021 IEEE/ACM 1st Workshop on AI Engineering-Software Engineering for AI (WAIN)*. IEEE, 121–128.

[153] Aniruddhan Murali, Gaurav Sahu, Kishanthan Thangarajah, Brian Zimmerman, Gema Rodríguez-Pérez, and Meiyappan Nagappan. 2024. Diversity in issue assignment: humans vs bots. *Empirical Software Engineering* 29, 2 (2024), 37.

[154] Giang Nguyen, Sumon Biswas, and Hridesh Rajan. 2023. Fix Fairness, Don't Ruin Accuracy: Performance Aware Fairness Repair using AutoML. *arXiv preprint arXiv:2306.09297* (2023).

[155] Eirini Ntoutsi, Pavlos Fafalios, Ujwal Gadiraju, Vasileios Iosifidis, Wolfgang Nejdl, Maria-Esther Vidal, Salvatore Ruggieri, Franco Turini, Symeon Papadopoulos, Emmanouil Krasanakis, et al. 2020. Bias in data-driven artificial intelligence systems—An introductory survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 10, 3 (2020), e1356.

[156] Julian Nyarko, Sharad Goel, and Roseanna Sommers. 2021. Breaking Taboos in Fair Machine Learning: An Experimental Study. In *Proceedings of the 1st ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization* (–, NY, USA) *(EAAMO '21)*. Association for Computing Machinery, New York, NY, USA, Article 14, 11 pages. https://doi.org/10.1145/3465416.3483291

[157] City of Chicago. Accessed: 16.01.2023. Strategic Subject List. https://data.cityofchicago.org/Public-Safety/Strategic-Subject-List/4aki-r3np

[158] U.S. Department of Labor. Accessed: 25.04.2022. Affirmative Action. https://www.dol.gov/general/topic/hiring/affirmativeact

[159] Moses Openja, Gabriel Laberge, and Foutse Khomh. 2023. Detection and evaluation of bias-inducing features in machine learning. *Empirical Software Engineering* 29, 1 (2023), 22.

[160] OpenSLR. Accessed: 16.01.2023. Crowdsourced high-quality nigerian english speech data set. http://openslr.org/70/

[161] Gourab K Patro, Lorenzo Porcaro, Laura Mitchell, Qiuyue Zhang, Meike Zehlike, and Nikhil Garg. 2022. Fair ranking: a critical review, challenges, and future directions. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*. 1929–1942.

[162] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2017. Deepxplore: Automated whitebox testing of deep learning systems. In *proceedings of the 26th Symposium on Operating Systems Principles*. 1–18.

[163] Kewen Peng, Joymallya Chakraborty, and Tim Menzies. 2022. FairMask: Better fairness via model-based rebalancing of protected attributes. *IEEE Transactions on Software Engineering* 49, 4 (2022), 2426–2439.

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

[165] Dana Pessach and Erez Shmueli. 2022. A review on fairness in machine learning. *ACM Computing Surveys (CSUR)* 55, 3 (2022), 1–44.

[166] Nga Pham, Hung Pham-Ngoc, and Anh Nguyen-Duc. 2023. Fairness Requirement in AI Engineering–A Review on Current Research and Future Directions. In *International Conference on Sustainability in Software Engineering & Business Information Management*. Springer, 3–13.

[167] Evaggelia Pitoura, Kostas Stefanidis, and Georgia Koutrika. 2022. Fairness in rankings and recommendations: an overview. *The VLDB Journal* (2022), 1–28.

[168] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. 2017. On fairness and calibration. *Advances in neural information processing systems* 30 (2017).

[169] Sai Sathiesh Rajan, Ezekiel Soremekun, Yves Le Traon, and Sudipta Chattopadhyay. 2024. Distribution-aware fairness test generation. *Journal of Systems and Software* 215 (2024), 112090. https://doi.org/10.1016/j.jss.2024.112090

[170] Sai Sathiesh Rajan, Sakshi Udeshi, and Sudipta Chattopadhyay. 2022. AequeVox: Automated Fairness Testing of Speech Recognition Systems. (2022).

[171] Stephen Rea. 2020. A Survey of Fair and Responsible Machine Learning and Artificial Intelligence: Implications of Consumer Financial Services. *Available at SSRN 3527034* (2020).

[172] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond Accuracy: Behavioral Testing of NLP Models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 4902–4912.

[173] Luke Rodriguez, Babak Salimi, Haoyue Ping, Julia Stoyanovich, and Bill Howe. 2018. MobilityMirror: Bias-adjusted transportation datasets. In *Workshop on Big Social Data and Urban Computing*. Springer, 18–39.

[174] Adam Rumpf and Hemanshu Kaul. 2021. A Public Transit Network Optimization Model for Equitable Access to Social Services. In *Proceedings of the 1st ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization* (–, NY, USA) *(EAAMO '21)*. Association for Computing Machinery, New York, NY, USA, Article 16, 17 pages. https://doi.org/10.1145/3465416.3483288

[175] Debjani Saha, Candice Schumann, Duncan C McElfresh, John P Dickerson, Michelle L Mazurek, and Michael Carl Tschantz. 2020. Human comprehension of fairness in machine learning. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 152–152.

[176] Pedro Saleiro, Benedict Kuester, Loren Hinkson, Jesse London, Abby Stevens, Ari Anisfeld, Kit T Rodolfa, and Rayid Ghani. 2018. Aequitas: A bias and fairness audit toolkit. *arXiv preprint arXiv:1811.05577* (2018).

[177] Babak Salimi, Bill Howe, and Dan Suciu. 2019. Data management for causal algorithmic fairness. *arXiv preprint arXiv:1908.07924* (2019).

[178] Babak Salimi, Bill Howe, and Dan Suciu. 2020. Database repair meets algorithmic fairness. *ACM SIGMOD Record* 49, 1 (2020), 34–41.

[179] Abel Salinas, Parth Shah, Yuzhong Huang, Robert McCormack, and Fred Morstatter. 2023. The Unequal Opportunities of Large Language Models: Examining Demographic Biases in Job Recommendations by ChatGPT and LLaMA. In *Proceedings of the 3rd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '23)*. Association for Computing Machinery, New York, NY, USA, Article 34, 15 pages.

https://doi.org/10.1145/3617694.3623257

[180] Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2020. Social Bias Frames: Reasoning about Social and Power Implications of Language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 5477–5490.

[181] Prasanna Sattigeri, Samuel C Hoffman, Vijil Chenthamarakshan, and Kush R Varshney. 2019. Fairness GAN: Generating datasets with fairness properties using a generative adversarial network. *IBM Journal of Research and Development* 63, 4/5 (2019), 3–1.

[182] Daniel Schwarcz. 2021. Health-Based Proxy Discrimination, Artificial Intelligence, and Big Data. *Artificial Intelligence, and Big Data (March 3, 2021). Houston Journal of Health Law and Policy* (2021).

[183] Meirav Segal, Anne-Marie George, and Christos Dimitrakakis. 2023. Policy Fairness and Unknown Bias Dynamics in Sequential Allocations. In *Proceedings of the 3rd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '23)*. Association for Computing Machinery, New York, NY, USA, Article 38, 10 pages. https://doi.org/10.1145/3617694.3623262

[184] Shahar Segal, Yossi Adi, Benny Pinkas, Carsten Baum, Chaya Ganesh, and Joseph Keshet. 2021. Fairness in the eyes of the data: Certifying machine-learning models. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*. 926–935.

[185] Arnab Sharma, Caglar Demir, Axel-Cyrille Ngonga Ngomo, and Heike Wehrheim. 2021. MLCheck-Property-Driven Testing of Machine Learning Models. *arXiv preprint arXiv:2105.00741* (2021).

[186] Shubham Sharma, Yunfeng Zhang, Jesús M Ríos Aliaga, Djallel Bouneffouf, Vinod Muthusamy, and Kush R Varshney. 2020. Data augmentation for discrimination prevention and bias disambiguation. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 358–364.

[187] Weijun Shen, Yanhui Li, Lin Chen, Yuanlei Han, Yuming Zhou, and Baowen Xu. 2020. Multiple-Boundary Clustering and Prioritization to Promote Neural Network Retraining. In *2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. 410–422.

[188] Shubham Singh, Bhuvni Shah, Chris Kanich, and Ian A. Kash. 2022. Fair Decision-Making for Food Inspections. In *Proceedings of the 2nd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '22)*. Association for Computing Machinery, New York, NY, USA, Article 5, 11 pages. https://doi.org/10.1145/3551624.3555289

[189] Kacper Sokol, Alexander Hepburn, Rafael Poyiadzi, Matthew Clifford, Raul Santos-Rodriguez, and Peter Flach. 2020. FAT forensics: a python toolbox for implementing and deploying fairness, accountability and transparency algorithms in predictive systems. *Journal of Open Source Software* 5, 49 (2020), 1904.

[190] Ezekiel Soremekun, Sakshi Udeshi, and Sudipta Chattopadhyay. 2022. Astraea: Grammar-based fairness testing. *IEEE Transactions on Software Engineering* (2022).

[191] Julia Stoyanovich. 2019. TransFAT: Translating fairness, accountability and transparency into data science practice. In *CEUR Workshop Proceedings*, Vol. 2417. CEUR-WS.

[192] Julia Stoyanovich, Serge Abiteboul, and Gerome Miklau. 2016. Data, responsibly: Fairness, neutrality and transparency in data analysis. In *International Conference on Extending Database Technology*.

[193] Youcheng Sun, Xiaowei Huang, Daniel Kroening, James Sharp, Matthew Hill, and Rob Ashmore. 2019. DeepConcolic: Testing and Debugging Deep Neural Networks. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. 111–114. https://doi.org/10.1109/ICSE-Companion.2019.00051

[194] Zeyu Sun, Zhenpeng Chen, Jie Zhang, and Dan Hao. 2024. Fairness Testing of Machine Translation Systems. *ACM Trans. Softw. Eng. Methodol.* 33, 6, Article 156 (June 2024), 27 pages. https://doi.org/10.1145/3664608

[195] Zeyu Sun, Jie M Zhang, Mark Harman, Mike Papadakis, and Lu Zhang. 2020. Automatic testing and improvement of machine translation. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 974–985.

[196] Zeyu Sun, Jie M Zhang, Yingfei Xiong, Mark Harman, Mike Papadakis, and Lu Zhang. 2022. Improving Machine Translation Systems via Isotopic Replacement. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. IEEE.

[197] Zeyu Tang, Jiji Zhang, and Kun Zhang. 2023. What-is and how-to for fairness in machine learning: A survey, reflection, and perspective. *Comput. Surveys* 55, 13s (2023), 1–37.

[198] Guanhong Tao, Weisong Sun, Tingxu Han, Chunrong Fang, and Xiangyu Zhang. 2022. RULER: discriminative and iterative adversarial training for deep neural network fairness. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 1173–1184.

[199] Philip S. Thomas, Bruno Castro da Silva, Andrew G. Barto, Stephen Giguere, Yuriy Brun, and Emma Brunskill. 2019. Preventing undesirable behavior of intelligent machines. *Science* 366, 6468 (2019), 999–1004. https://doi.org/10.1126/science.aag3311 arXiv:https://www.science.org/doi/pdf/10.1126/science.aag3311

[200] Yuchi Tian, Ziyuan Zhong, Vicente Ordonez, Gail Kaiser, and Baishakhi Ray. 2020. Testing DNN image classifiers for confusion & bias errors. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 1122–1134.

[201] Saeid Tizpaz-Niari, Ashish Kumar, Gang Tan, and Ashutosh Trivedi. 2022. Fairness-aware Configuration of Machine Learning Libraries. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. IEEE.

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

[203] Sakshi Udeshi, Pryanshu Arora, and Sudipta Chattopadhyay. 2018. Automated directed fairness testing. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*. 98–108.

[204] Sahil Verma and Julia Rubin. 2018. Fairness definitions explained. In *2018 ieee/acm international workshop on software fairness (fairware)*. IEEE, 1–7.

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

[206] Junjie Wang, Ye Yang, Song Wang, Jun Hu, and Qing Wang. 2022. Context- and Fairness-Aware In-Process Crowdworker Recommendation. *ACM Trans. Softw. Eng. Methodol.* 31, 3, Article 35 (mar 2022), 31 pages. https://doi.org/10.1145/3487571

[207] Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. 2019. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 5310–5319.

[208] Zeyu Wang, Klint Qinami, Ioannis Christos Karakozis, Kyle Genova, Prem Nair, Kenji Hata, and Olga Russakovsky. 2020. Towards fairness in visual recognition: Effective strategies for bias mitigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8919–8928.

[209] Zhaohui Wang, Min Zhang, Jingran Yang, Bojie Shao, and Min Zhang. 2024. MAFT: Efficient Model-Agnostic Fairness Testing for Deep Neural Networks via Zero-Order Gradient Search. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 1–12.

[210] Nimmi Rashinika Weeraddana, Xiaoyan Xu, Mahmoud Alfadel, Shane McIntosh, and Meiyappan Nagappan. 2023. An empirical comparison of ethnic and gender diversity of DevOps and non-DevOps contributions to open-source projects. *Empirical Software Engineering* 28, 6 (2023), 150.

[211] Steven H Weinberger and Stephen A Kunath. 2011. The Speech Accent Archive: towards a typology of English accents. In *Corpus-based studies in language use, language learning, and language documentation*. Brill, 265–281.

[212] Benjamin Wilson, Judy Hoffman, and Jamie Morgenstern. 2019. Predictive inequity in object detection. (2019).

[213] Yongkai Wu, Lu Zhang, Xintao Wu, and Hanghang Tong. 2019. Pc-fairness: A unified framework for measuring causality-based fairness. *Advances in Neural Information Processing Systems* 32 (2019).

[214] Yisong Xiao, Aishan Liu, Tianlin Li, and Xianglong Liu. 2023. Latent Imitator: Generating Natural Individual Discriminatory Instances for Black-Box Fairness Testing. *arXiv preprint arXiv:2305.11602* (2023).

[215] Ying Xiao, Jie M Zhang, Yepang Liu, Mohammad Reza Mousavi, Sicen Liu, and Dingyuan Xue. 2024. MirrorFair: Fixing fairness bugs in machine learning software via counterfactual predictions. *Proceedings of the ACM on Software Engineering* 1, FSE (2024), 2121–2143.

[216] Xiaofei Xie, Lei Ma, Felix Juefei-Xu, Minhui Xue, Hongxu Chen, Yang Liu, Jianjun Zhao, Bo Li, Jianxiong Yin, and Simon See. 2019. DeepHunter: A Coverage-Guided Fuzz Testing Framework for Deep Neural Networks. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis* (Beijing, China) *(ISSTA 2019)*. Association for Computing Machinery, New York, NY, USA, 146–157. https://doi.org/10.1145/3293882.3330579

[217] Junjie Yang, Jiajun Jiang, Zeyu Sun, and Junjie Chen. 2024. A large-scale empirical study on improving the fairness of image classification models. In *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*. 210–222.

[218] Liu Yang, Mingyang Zhang, Cheng Li, Michael Bendersky, and Marc Najork. 2020. Beyond 512 tokens: Siamese multi-depth transformer-based hierarchical encoder for long-form document matching. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1725–1734.

[219] 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*. 1540–1544.

[220] Christina Yeung, Umar Iqbal, Tadayoshi Kohno, and Franziska Roesner. 2023. Gender Biases in Tone Analysis: A Case Study of a Commercial Wearable. In *Proceedings of the 3rd ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '23)*. Association for Computing Machinery, New York, NY, USA, Article 21, 12 pages. https://doi.org/10.1145/3617694.3623241

[221] Jun Yu, Xinlong Hao, Haonian Xie, and Ye Yu. 2020. Fair face recognition using data balancing, enhancement and fusion. In *European Conference on Computer Vision*. Springer, 492–505.

[222] Zhe Yu, Joymallya Chakraborty, and Tim Menzies. 2024. FairBalance: How to Achieve Equalized Odds With Data Pre-Processing. *IEEE Trans. Softw. Eng.* 50, 9 (Sept. 2024), 2294–2312. https://doi.org/10.1109/TSE.2024.3431445

[223] Meike Zehlike, Ke Yang, and Julia Stoyanovich. 2021. Fairness in Ranking: A Survey. *arXiv preprint arXiv:2103.14000* (2021).

[224] Meike Zehlike, Ke Yang, and Julia Stoyanovich. 2022. Fairness in ranking, part ii: Learning-to-rank and recommender systems. *Comput. Surveys* 55, 6 (2022), 1–41.

[225] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning fair representations. In *International conference on machine learning*. PMLR, 325–333.

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

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

[228] Lingfeng Zhang, Yueling Zhang, and Min Zhang. 2021. Efficient white-box fairness testing through gradient search. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*. 103–114.

[229] Mengdi Zhang and Jun Sun. 2022. Adaptive fairness improvement based on causality analysis. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 6–17.

[230] Mengdi Zhang, Jun Sun, Jingyi Wang, and Bing Sun. 2023. TestSGD: Interpretable Testing of Neural Networks against Subtle Group Discrimination. *ACM Trans. Softw. Eng. Methodol.* 32, 6, Article 137 (sep 2023), 24 pages. https://doi.org/10.1145/3591869

[231] Peixin Zhang, Jingyi Wang, Jun Sun, Guoliang Dong, Xinyu Wang, Xingen Wang, Jin Song Dong, and Ting Dai. 2020. White-box fairness testing through adversarial sampling. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 949–960.

[232] Peixin Zhang, Jingyi Wang, Jun Sun, Xinyu Wang, Guoliang Dong, Xingen Wang, Ting Dai, and Jin Song Dong. 2021. Automatic Fairness Testing of Neural Classifiers through Adversarial Sampling. *IEEE Transactions on Software Engineering* (2021).

[233] Xueru Zhang and Mingyan Liu. 2021. Fairness in learning-based sequential decision algorithms: A survey. In *Handbook of Reinforcement Learning and Control*. Springer, 525–555.

[234] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. *arXiv preprint arXiv:1707.09457* (2017).

[235] Haibin Zheng, Zhiqing Chen, Tianyu Du, Xuhong Zhang, Yao Cheng, Shouling Ji, Jingyi Wang, Yue Yu, and Jinyin Chen. 2022. NeuronFair: Interpretable White-Box Fairness Testing through Biased Neuron Identification. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. IEEE.

[236] Wei Zheng, Lidan Lin, Xiaoxue Wu, and Xiang Chen. 2024. An Empirical Study on Correlations Between Deep Neural Network Fairness and Neuron Coverage Criteria. *IEEE Trans. Softw. Eng.* 50, 3 (March 2024), 391–412. https://doi.org/10.1109/TSE.2023.3349001

[237] W. Zheng, L. Lin, X. Wu, and X. Chen. 5555. An Empirical Study on Correlations between Deep Neural Network Fairness and Neuron Coverage Criteria. *IEEE Transactions on Software Engineering* 01 (jan 5555), 1–22. https://doi.org/10.1109/TSE.2023.3349001

[238] Indre Žliobaite, Faisal Kamiran, and Toon Calders. 2011. Handling conditional discrimination. In *2011 IEEE 11th International Conference on Data Mining*. IEEE, 992–1001.