Learn2Discover: An Active Learning Approach to Automated Fairness Testing in AI

Abigail Lithwick
Ayesha Ali
Lincoln Briscoe
Matthew Alix
Pooja Kayyalathodi
Teigan Rosen

***Abstract*—Today, artificial intelligence (AI) systems are being used increasingly for important decision-making problems, such as criminal justice cases, which have the potential to affect human outcomes. It is essential that these systems do not perpetuate bias, and as such, developing methods for testing the fairness of AI systems has become critical. State-of-the-art fairness testing approaches, including Themis and AEQUITAS, have focused on automatically generating test cases in order to identify and mitigate bias. However, current approaches to fairness testing are each restricted to different definitions of fairness, causing the overall consensus on what constitutes a biased decision to be inconclusive. Furthermore, the alternative to existing approaches requires testers to manually label a large number of test cases as either fair or unfair, which demands a high amount of human effort.**

Our solution to these gaps, termed *Learn2Discover*, utilises a machine learning model trained via active learning to query the user about test cases that it is uncertain about. In doing so, it learns how to identify and correctly label cases of unfairness. Through the incorporation of domain context in training, as well as minimising human effort, we anticipate that our work will contribute to methods that expedite the development of fairer and more explainable AI. Our results demonstrate that our approach is successfully able to predict cases of unfairness in fairness testing datasets with high precision. Finally, we show that Learn2Discover has can successfully reduce the human effort required to produce a fairness oracle with meaningful results.

I. Introduction

Machine learning (ML) is a rapidly growing field of research. Today, increasingly complex artificial intelligence (AI) systems empower software to make decisions autonomously. This software is now being used in many scenarios that directly affect human outcomes, including hiring decisions [1], criminal sentencing [2] [3] [4], and medical risk assessment [5].

While the increased uptake of machine learning systems for decision-making leads to an increase in efficiency, there is also an associated risk of preserving historical biases. Thus, what was once a suggestion of evaluating the fairness of these systems has now become an imperative — it is crucial that these decision-making systems are seen to be fair, and not influenced by unexplainable bias. Fairness testing has emerged as a means of remedying such biases.

For instance, consider an AI system that makes decisions regarding loan applications. While this may seem like a simple application on the surface, loan approvals have been historically affected by motivated bias against protected attributes, including ethnicity, gender, and sexual orientation [6] [7] [8] [4]. Here, fairness testing can have a role in reducing the risks that such systems are deployed [9] by evaluating the decisions that they make.

Current fairness testing approaches — including Themis [10], AEQUITAS [11], and NeuronFair [12] — focus on test case generation and discrimination detection. However, these approaches rely on narrow formulations of fairness which are evaluated with respect to explicitly predefined sensitive attributes. In contrast, our approach, Learn2Discover, aims to remove definition dependence while minimising human effort in labelling test cases. By reducing human effort, we aim to expedite AI development time while simultaneously improving decision fairness.

Learn2Discover aims to train an oracle to determine whether an output, given a specific input, is fair. To achieve this, we have implemented an active learning loop which aims to predict whether a human oracle would label the result of a test case as fair or unfair, with a particular focus on identifying cases of unfairness.

Importantly, Learn2Discover decides which test cases require human labelling based on what would be most informative for improving its model, with the aim of minimising the number of test cases presented. In this way, we intend for Learn2Discover to be a stepping stone for future developments in AI fairness testing.

II. Repository

Learn2Discover can be found at https://github.com/mjcalx/learn2discover

III. Definitions

The following are definitions pertinent to the discussion of Learn2Discover:

Definition 1. *Discrimination* is a difference that occurs when a change in inputs results in different outcomes.

Some examples of discrimination are:

1) Two people apply for a loan. Person A has a higher yearly income than person B. The result of the applications is that person A is accepted but person B is denied.

2) Another example is that person A and B both have the same salary, however person A is younger than person B, and person A's application is rejected.

Definition 2. *Bias* is wrongful or unjustifiable discrimination.

Referring back to the loan application example in the introduction, an example of bias is: if a person is rejected a loan solely based on their ethnicity this can be considered a biased outcome as the discrimination here is unjustifiable.

Definition 3. *A Sensitive Attribute* is an attribute that is considered to have a significant impact on an unfair decision.

An example of a sensitive attribute is: a system is known to hire males more often than females. In this case, gender would be a sensitive attribute and the system has learned to discriminate against this sensitive attribute.

Definition 4. *Fairness.* An outcome is *fair* if it is free of bias.

With reference to Definition 1, Example 1, this outcome may be considered fair, as the person with the lower salary may not be able to pay off the loan. However, Definition 1, Example 2 would be considered unfair, as age alone does not affect a person's ability to pay off their loan.

Definition 5. *A System* is a function that, given a set of input attributes, returns a set of output attributes after following some logical process.

In the context of Learn2Discover, a system can be anything from a software system to a real-world judicial system. Importantly, if given the same set of input values, the system should return the same set of output values. For the purposes of our research, representations of systems are abstracted to finite sets of data instances, each consisting of a set of input values, and the system's corresponding set of output values.

Definition 6. *The System Under Test (SUT)* is the system being evaluated by an oracle

In the context of our research, that oracle will be Learn2Discover.

IV. Literature Review

A. State-of-the-Art Fairness Testing

Fairness testing is an approach to testing ML systems by generating test cases to detect, identify and ultimately help to eradicate unfairness. One of the earliest recognised approaches to fairness testing is Themis [10], which defines AI software as a black box, and generates a set of test cases to reveal both group and causal discrimination. While Themis was able to successfully detect discrimination, their approach was found to be somewhat inefficient, generating many duplicate test cases. As such, AEQUITAS was proposed, which aimed to highlight individual fairness violations using a probabilistic approach [11], [12]. AEQUITAS had the capability to efficiently generate test cases, causing less duplicates to arise, and leverage these to improve fairness. However, like Themis, their work was not able to identify the cause of discrimination in the inputted models. The same year, Agarwal et. al. [13] utilised symbolic execution and local explainability to systematically generate test inputs to detect individual discrimination. This approach was once again successful in test case generation, and their use of local explanation allowed users to understand each prediction of the model. All three of the aforementioned approaches to fairness testing, however, were black box testing approaches, and none of them were directly applicable to deep neural networks. This left a gap for white-box testing, which was addressed by NeuronFair [12], a quantitative white-box fairness testing approach used to detect and interpret instances of unfairness in a system. NeuronFair also extended the scope of their research to include unstructured data including image, text, and speech, which earlier models did not handle.

B. Problem definition

The above-mentioned methods are mostly designed to generate test cases to detect unfairness based on certain predefined sensitive attributes. This may be helpful in scenarios where certain attributes can be confidently isolated from others in terms of their impact on fairness, however in many cases it will oversimplify the problem, causing key context points to be lost and unconsidered.

Furthermore, while these approaches are very generalisable to different systems, they are restricted to specific definitions of fairness. There are three main categories of fairness that these definitions fall into: Individual, Group, and Subgroup fairness. These formulations are defined by Meharabi et al. as: [4]

Individual Fairness. When similar predictions are given to similar individuals.

Group Fairness. When different groups are treated equally.

Subgroup Fairness. A hybrid approach combining Individual and Group Fairness and aiming to improve on both, taking a large group fairness constraint and evaluating whether it holds over a large collection of subgroups.

Current literature has not converged on a specific definition of fairness [4]; as stated above definitions may be mutually exclusive [14], and in some cases the preferred formulation is subject to change [15]. This lack of consistency between the above approaches means that each approach may return a different set of results.

C. Active Learning

Active learning is a semi-supervised type of machine learning approach, in which the system retains control and humans are used as oracles to label the data [16]. Active learning uses an iterative process to receive data. This means that labelled data is not provided all at once; rather, the oracle requests information based on a query strategy. For example, the oracle may query the human for the labels that it is least confident about. Active learning is a method which has been used in various scenarios, such as recognising multiple plankton types with images [17], and classifying cancer pathology reports [18]. The main benefit of active learning is that retrieving unlabelled data is less expensive than labelled data. As such, an active learning model will often require a smaller number of training instances while achieving a similar level of accuracy. Active learning is

therefore especially useful when training is extremely costly or time consuming [16].

In the case of fairness testing, test cases are traditionally labelled manually by one or more testers. To achieve effectiveness in the fairness testing model, it is important to have a large and high-quality dataset, and thus labelling test cases can require a high degree of human effort. Here, fairness testing can benefit greatly from an active learning approach, as the human effort required to label test cases may be reduced while maintaining comparable accuracy [19].

Active learning has an additional benefit to fairness testing: it helps to introduce human context to the labelling process. Existing automated fairness testing approaches, while requiring very little human effort, fail to include human context in their labelling schemata. Human context is important to understanding fairness, especially in domains with high stakes like healthcare, education and public law enforcement. Many researchers have examined in various contexts how decisions made by software are perceived by humans. For example, Longoni et. al. [20] researched consumer receptivity to algorithmic decisions in regards to diagnosis and treatment. They found that people tend to be more resistant to algorithmic decisions than those made by humans, and the main reasoning provided was that algorithms cannot account for an individual's unique symptoms and circumstances. Additionally, Lee and Baykal [21] had researched people's perception of algorithmic decisions with respect to a website named Spliddit, which assists in social decisions such as dividing household rent, chores, credit and goods. The results indicated that people viewed algorithmic decisions as unfair one third of the time, citing that decisions failed to account for group dynamics and other social factors. Ultimately, research indicates that having a human-in-the-loop active learning approach reduces resistance to decisions made by software systems [22].

D. Automated Test Oracles

In the field of software testing, the *oracle problem* is defined as the challenge of distinguishing the desired behaviour of a software system from potentially incorrect behaviours [23]. A *test oracle* is defined as a system that is used to solve the test oracle problem. The simplest test oracle is a human, who understands the software and can identify faulty or undesirable behaviour. While a human can make helpful contributions, there are significant costs associated with using human oracles in lieu of an automated approach [24] [25] [26].

Oliveira et al. [27] identifies three main types of test oracles: *implicit*, *specified*, and *derived* oracles. Derived oracles use information derived from various data sources to create the test oracle [23]. Since we aim to use generalised datasets and live user inputs in an active learning loop, our research will implement a derived oracle. A recent survey [28] determined that existing work on fairness testing employs only one of two types of derived test oracles: using either metamorphic [10] or statistical relations as test oracles [12]. To date, we find that no existing fairness testing research utilises a human-in-the-loop as a part of the test oracle.

V. Research Questions and Expected Results

Learn2Discover aims to reduce the human effort involved in fairness testing, while maintaining the ability to correctly identify unfairness. Learn2Discover has been designed to target AI or machine learning-based software systems that are used to make decisions. It aims to evaluate the fairness of these decisions with respect to the annotations of a human or equivalent oracle.

In developing Learn2Discover, we focused on two main research questions:

RQ1. How much can Learn2Discover reduce the human effort involved in labelling test cases? We aim to reduce overall human effort involved in the process of fairness testing, by minimising the number of test cases that a human oracle needs to label while maintaining an effective strategy for identifying unfairness.

RQ2. How effective is Learn2Discover in identifying unfairness? When evaluating our approach's effectiveness, we will consider the metrics of accuracy, precision, recall, and F1-score. The emphasis will be on precision, because the main priority of this research is to minimise false positives. We have formulated two sub-questions to help us understand the performance of Learn2Discover:

> **RQ2.1. How does the choice of query strategy impact the active learning loop's performance?** Our active learning loop must select which test cases to display to the user using a query strategy [29]. There are two common query strategies which will be considered and compared for Learn2Discover:
>
> > *Least confidence*: the active learning loop selects the test cases whose best labelling it is least confident about [29]
> >
> > *Entropy sampling*: the active learning loop uses an information-theoretic measure called entropy in order to select the test cases that it is least confident about [29]
>
> **RQ2.2. How does the choice of stopping criterion impact the active learning loop's performance?** We will explore how the performance of Learn2Discover is affected by the following factors: *confidence*, *number of test cases shown*, or *number of iterations*. While confidence is likely the most difficult stopping criteria to calculate, it is often the most robust stopping criteria.

To answer these research questions, we plan to carry out multiple independent runs of the active learning loop under a variety of conditions, and consider the resulting evaluation metrics.

VI. Research Methodology

A. Input Data Generation

Learn2Discover requires each data instance in its input dataset to be assigned a fairness label from the set $\{\texttt{fair}, \texttt{unfair}\}$. However, we were unable to find any readily available data that met this requirement. Additionally, existing fairness testing tools were also found to be incapable of providing us with these labels, as their outputs describe

the relative fairness between sets of data instances [11] [10] [30]. In response, we developed a new method to acquire the necessary training data.

This method takes as input a dataset D with attributes of the form $\{X, Y\}$ from some system, where X is a set of input attributes and Y is a set of output attributes. Our data generation approach then proceeds as described by Algorithm 1.

Learn2Discover is ultimately intended to evaluate the fairness of unit test outcomes in AI systems. To align with this goal, we will model each data instance $d \in D$ as the result of a unit test, with inputs and a binary $\{0, 1\}$ outcome. To achieve this, we define for each dataset, given $d \in D$:

- $val(x, d) = \{\text{The value of attribute } x \text{ in data instance } d\}$
- $f : y(d) \to \{0, 1\},\ y(d) = \{\gamma \in Y : val(\gamma, d)\}$

where $f(y(d))$ maps a data instance's output values to a binary $\{0, 1\}$ outcome.

To give a simple example, the COMPAS dataset [3] contains the output attribute $\mathrm{ScoreText}$, with possible values $\{\mathrm{Low}, \mathrm{Medium}, \mathrm{High}\}$. One possible outcome function for this dataset is:

$$f(y(d)) = \begin{cases} 0 & \text{if } y(d)[\mathrm{ScoreText}] = \mathrm{Low} \\ 1 & \text{if } y(d)[\mathrm{ScoreText}] \in \{\mathrm{Medium}, \mathrm{High}\} \end{cases}$$

To determine the final fairness labels, a simple fairness oracle will need to be implemented. This oracle should either produce the final fairness label directly, or should output a numeric score which can be used to determine the final labelling. Additionally, the oracle should consider both the input attributes and outcome when determining its fairness result, as fairness is, by definition, determined with respect to discrimination, which itself is also determined with respect to inputs and their corresponding outputs.

The oracle we have chosen to implement is a Group Fairness Oracle which, given a set of sensitive attributes $S \subseteq X$, counts for each distinct value $v = val(u, d),\ u \in S,\ d \in D$:

- The number of data instances $c(u, v)$, where attribute u has value v
- The number of data instances $p(u, v)$, where attribute u has value v and $f(y(d)) = 0$

For each v, we then compute a group fairness score $q(v) = \frac{c(u,v)}{p(u,v)}$. Since $0 < p(u, v) \leq c(u, v)$, we have $0 < q(v) \leq 1$, with the intuition that values of $q(v)$ closer to 0 are less fair, and values closer to 1 are more fair.

To augment this intuition, we then normalise each $q(v)$ to a new value $\omega(v)$, where $-1 \leq \omega(v) \leq 1$. This gives us the new intuition of:

$$\begin{cases} \omega(v) < 0 \implies v \text{ contributes to an unfair outcome} \\ \omega(v) > 0 \implies v \text{ contributes to a fair outcome} \end{cases}$$

Finally, for each data instance $d \in D$, we compute the sum of normalised fairness scores:

$$z(d) = \sum_{u \in S} \omega(val(u, d))$$

Algorithm 1 Determine Fairness Labels (Group Fairness oracle)

Require: $D = \{\mathrm{DataInstances}\}$ ▷ Input dataset
Require: $S \subseteq X$ ▷ Sensitive attributes
Require: $f : y(d) \to \{0, 1\}$ ▷ Outcome function
for all $d \in D$ **do** ▷ Determine 0-1 outcomes
  outcome $\leftarrow f(y(d))$
end for

for all $u \in S$ **do**
  $V_u \leftarrow \{\}$
  for all $d \in \mathrm{D}$ **do** ▷ Find distinct sensitive values
    if $val(u, d) \notin V_u$ **then**
      $V_u \leftarrow V_u \cup val(u, d)$
    end if
  end for

  for all $v \in V_u$ **do** ▷ Compute group fairness scores
    $c_{u,v} \leftarrow ||\{d \in D : val(u, d) = v\}||$
    $p_{u,v} \leftarrow ||\{d \in D : val(u, d) = v,\ f(y(d)) = 0\}||$
    $q_v \leftarrow \frac{c_{u,v}}{p_{u,v}}$
    $\omega_v \leftarrow \mathrm{normalise}(q_v)$
  end for
end for

for all $d \in D$ **do** ▷ Determine final fairness labels
  $z_d \leftarrow \sum_{u \in S} \omega(val(u, d))$
  if $z_d \geq 0$ **then**
    $\lambda_d \leftarrow \mathrm{fair}$
  else
    $\lambda_d \leftarrow \mathrm{unfair}$
  end if
  $d[\mathrm{FairnessLabel}] \leftarrow \lambda_d$
end for

We then determine the final fairness label $\lambda(d)$ as follows:

$$\lambda(d) = \begin{cases} \mathrm{fair} & \text{if } z(d) \geq 0 \\ \mathrm{unfair} & \text{if } z(d) < 0 \end{cases}$$

Continuing from our previous example, suppose using the COMPAS dataset [3], we define sensitive attributes to be $\{\mathrm{sex}, \mathrm{ethnicity}\}$, with values:

- $val(\mathrm{sex}, d) \in \{\mathrm{male}, \mathrm{female}\}$
- $val(\mathrm{ethnicity}, d) \in \{\mathrm{caucasian}, \mathrm{african\text{-}american}, \mathrm{hispanic}\}$

Consider now a single data instance $a \in D$, with $val(\mathrm{sex}, a) = \mathrm{female},\ val(\mathrm{ethnicity}, a) = \mathrm{african\text{-}american}$. Suppose now that of the 60798 total data instances, 13319 are female, and of the female data instances, 9796 have 0 outcomes. The unnormalised group fairness score will therefore be $q(\mathrm{female}) = \frac{9796}{13319} \approx 0.735$. Performing a similar calculation, we find that $q(\mathrm{african\text{-}american}) \approx 0.523$.

Now suppose that these scores are then normalised to $\omega(\mathrm{female}) \approx 0.034,\ \omega(\mathrm{african\text{-}american}) \approx -0.144$. This makes the sum of normalised fairness scores for this data instance $z(a) \approx -0.11$, and so the final fairness label will be $\lambda(a) = \mathrm{unfair}$.

Algorithm 2 Learn2Discover

Input: $unlabelled\_data$, $test\_data$
1: **global variables**
2: $model \leftarrow$ L2DClassifierInit()
3: $qstrat \leftarrow$ QueryStrategyInit()
4: $stop \leftarrow$ StoppingCriterionInit()
5: $training\_data \leftarrow \emptyset$
6: $unlabelled\_data$, $test\_data$
7: **end global variables**
8:
9: **procedure** Run()
10: **while** $stop$.Condition() is False **do**
11: ActiveLearningLoop()
12: **end while**
13: $model$.Evaluate($test\_data$)
14: **end procedure**
15:
16: **procedure** ActiveLearningLoop()
17: $sample \leftarrow qstrat$.Query($unlabelled\_data$)
18: $annotated \leftarrow$ Annotate($sample$)
19: UpdateTrainingData($annotated$)
20: $model$.Train($training\_data$)
21: **end procedure**
22:
23: **procedure** UpdateTrainingData(new_data)
24: $training\_data \leftarrow training\_data \cup new\_data$
25: $unlabelled\_data \leftarrow unlabelled\_data - new\_data$
26: **end procedure**

B. Experimental Protocol

Our Learn2Discover implementation begins its execution by loading a fairness-labelled dataset, as generated in Algorithm 1. The instances are of the form $((t, y'), l) \in \mathbb{I}$, where

- t is the input given to the SUT
- y' is a binary outcome as determined by the SUT
- $l \in L$; $L = \{\texttt{fair}, \texttt{unfair}\}$
- $\mathbb{I}$ is the input space

This data is then split into test, validation, and unlabelled sets. Note that, since our data generation implementation uses an attached oracle to supply fairness labels, the learner ignores the labels in the unlabelled set and treats the instances as unlabelled. Learn2Discover then proceeds as shown in Algorithm 2.

Our method utilises an active learning loop to train an ML model to predict a fairness label on an SUT outcome. Prior to training, a query strategy Q and stopping criterion P are chosen (lines 3 and 4), in accordance with *RQ1* and *RQ2* respectively. We then iterate over the active learning loop, an outline of which is shown from line 16.

The learner first applies Q to the unlabelled set $U \subseteq \mathbb{I}$ to produce a sample $S' = Q(U) = \{t_i : t_i \text{ is chosen by } Q\}$ of instances which will be the most informative to the learner for improving its model (line 17). The model itself may be used to inform this query, though its outputs are not used for further training of the model while being queried. The oracle is then prompted to supply fairness label annotations for S' (line 18). The training data and unlabelled data are then updated with respect to the annotated instances (line 19), and we train the model on the updated training data $(T \leftarrow T \cup S') \subseteq \mathbb{I}$.

During training, the learner applies its current model to T, and for each instance $t_i \in T$ outputs a probability distribution over L. The loss of the highest-confidence label against the actual label is then measured and used to optimise the model.

This loop—query, then annotate, then train—is repeated until P is satisfied. After the loop terminates, the test data are fed to the model and the evaluation statistics are calculated.

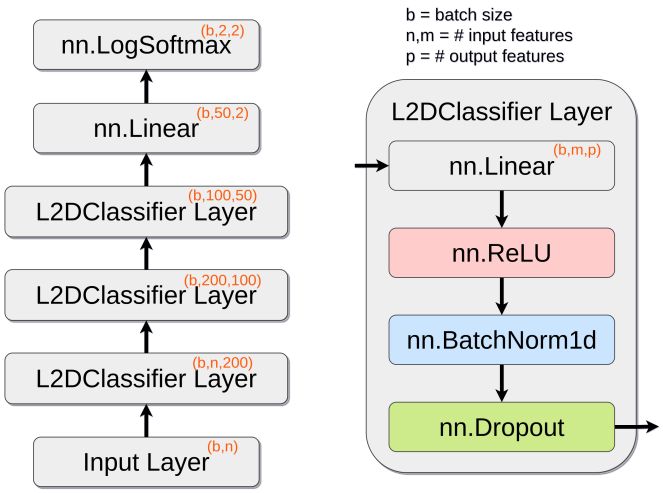

Fig. 1. Architecture of `L2DClassifier`

C. Structure of the Learner

The architecture of the model to be trained (Algorithm 2, line 20) is outlined in Figure 1. For this project we have created a feedforward neural network using stochastic gradient descent optimisation, with negative log-likelihood as the loss function. These design choices were made due to their longevity in the literature and historical success in classification tasks. [31] [32] [33]

The L2DClassifier layer structure contains standard elements found in other classification networks:

- *ReLU (rectified linear unit)* is the activation function [34]

$$g(x) = \max(0, x) = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases}$$

  where x is the input to the activation layer of the network. For our purposes, x is the output of a linear transformation on the model inputs, as reflected in Figure 1. ReLU has proved to be a popular choice among contemporary neural networks [35]; it is more resilient to vanishing gradients than other activation functions, and is more computationally efficient than sigmoid-family functions [34].
- *Batch Normalisation* is a technique used to minimise internal covariate shift, allowing higher learning rates and

faster overall training [36]. It does this by normalising layer inputs to a standard distribution:

$$x' = \frac{x - \mathbb{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta$$

where x is the input to the batch norm layer, γ and β are parameters learned during training, and ϵ is a small constant.

- *Dropout* is regularisation technique which (1) randomly zeros elements of the input with some preconfigured probability p, and (2) scales outputs by a factor of $\frac{1}{1-p}$. The dropout layer is then disabled during evaluation. This technique enables sparser and more generalisable representations to be learned, by preventing co-adaptation between neurons [37].

All of the components described were imported from the pyTorch[1] library.

D. Query Strategies

Two query strategies were implemented for this project:

Least Confidence. Samples selected minimise the output probability of the highest-confidence binary fairness label, normalised to a range of $[0.5, 1]$.

Entropy. Samples selected maximise the Shannon entropy [38]

$$H(L) = -\sum_{l \in \mathcal{L}} p(l) \log_2 p(l) = \mathbb{E}[-\log_2 p(L)]$$

of the highest-confidence binary fairness label, where $\mathcal{L}$ is the set of output labels, and L is a discrete random variable distributed according to $p : \mathcal{L} \to [0, 1]$.

As fairness labelling is currently binary (`fair` or `unfair`) we would expect the entropy strategy to perform identically to the least confidence strategy, on average. This is because we expect both strategies to return the same set of instances for a given sample. That is,

$$\text{Confs} = \{\hat{y}_i : \hat{y}_i = \arg\max_{l_j}\{p(l_j)\},\ l_j \in \mathcal{L},\ i \leq n\}$$

$$\text{Entropies} = \{H(\hat{y}_i) : i \leq n\}$$

where n is the size of the sample. We easily see that the maximum entropy for any $\hat{y}_i$ is 1, which corresponds to $p(l_j) = 0.5$, the minimum possible confidence. The same relationship holds with the minimum entropy and maximum confidence. However, we would expect the implemented strategies to differ in performance when used in a task with more than two fairness labels.

E. Stopping Criteria

Three stopping criteria were implemented for this project. The truth value of the selected criterion is evaluated at line 10 of Algorithm 2

Number of loop iterations. The loop is repeated for a fixed number of iterations.

Number of annotations. The loop is repeated until a threshold of annotated examples has been reached. The number of iterations will vary based on the preconfigured sample size of unlabelled data.

Confidence threshold. The loop is repeated until a threshold of mean confidence on the validation set has been reached.

F. Evaluation

To answer *RQ1*, human effort will be measured in terms of the number of annotations (labels) queried from the human or equivalent oracle. We will plot confidence, accuracy and precision across 250 annotations for all datasets. This value was chosen as, in practice, human oracles would be expected to label significantly less than 250 test cases. As such, this number allows us to make observations about the ideal number of annotations for our system.

Given this data, we will determine a number of annotations a at which the precision exceeds a pre-defined threshold, and all subsequent annotations $a+k, k \in \mathbb{N}$ yield precision values above the threshold. This threshold was selected to be 0.8, as it gives a good balance between a high precision, and minimising a. We will consider a to represent the human effort required for a given dataset.

To evaluate the reduction in human effort achieved we consider the final precision p, and the stopping number of annotations a. Additionally, we must consider the total number of data instances T in the dataset.

Precision was chosen as the key metric for measuring human effort; fairness testing aims to correctly identify unfair test cases to mitigate biased outcomes in the underlying system. Therefore, in discussing human effort, we consider the precision to represent the effectiveness of our approach.

The expected number of correctly identified unfair outcomes can be evaluated by: $F \leq p(T - a) + a$. That is, in a dataset of size T, we expect to correctly identify at most F unfair test cases with Learn2Discover's assistance.

Compared to the case where a human must manually label all F test cases, if F is maximal, then the expected reduction in human effort due to Learn2Discover is: $R = 1 - \frac{a}{F}$.

To answer *RQ2*, accuracy, precision, recall, and F1-score will be measured for different combinations of stopping criteria and query strategies. We aim to compare outcomes for different configurations of Learn2Discover in order to determine the answers to the sub-questions *RQ2.1* and *RQ2.2*. Finally, we determine the ideal combination of query strategy and stopping criterion to answer *RQ2*.

In Learn2Discover:

- A **positive** test case is a test case that is labelled as unfair. This assignment was chosen as Learn2Discover focuses on identifying cases of unfairness, in order to ultimately mitigate bias in a system. While still important, it is less critical to Learn2Discover that fair test cases are correctly identified.
- A **negative** test case a test case that is labelled as fair.

[1]https://pypi.org/project/torch/

- **Accuracy** refers to the proportion of correctly classified test cases, with respect to the total number of test cases. Note that if the model is disproportionately successful at identifying a label l and the dataset has a large proportion of test cases labelled l, then accuracy can be unexpectedly high. For this reason, accuracy will not be discussed at length.
- **Precision** refers to the proportion of true positives, with respect to the total number of positives identified by Learn2Discover. Thus, if Learn2Discover has a precision of 0.8, then we expect 80% of test cases labelled unfair by Learn2Discover to be correctly identified.
- **Recall** refers to the proportion of true positives predicted by the model, with respect to the total number of positive test cases present in the evaluation dataset. Thus, if Learn2Discover has a recall of 0.8, then 80% of the test cases expected to be labelled unfair were correctly identified by Learn2Discover. Note that it is usually impossible to achieve both high precision and recall [39].
- **F1 score** refers to the weighted average of precision and recall. It can be a better representation of the true effectiveness of a model compared to accuracy in cases where the ratio of true positives to true negatives is disproportionate.

VII. Results

Experimental setup. We evaluate Learn2Discover across a variety of datasets, as outlined in Table I. These datasets have been used widely in fairness testing research [10] [11] [40]. The largest datasets, COMPAS and Adult, had 10% of their data instances allocated for training data. However, the two smaller datasets required a larger proportion of training data, at 35% of their data instances in order to provide a sufficient amount of data for labelling. The data shown in Table I was calculated after appropriate pre-processing was applied, which involved removing redundant attributes and test cases with unknown or empty values, and collecting numeric data (such as age) into categories. The sensitive attributes and fair:unfair ratio are defined as part of our data generation approach, and are not inputted directly into Learn2Discover.

For each dataset, Learn2Discover was run and evaluated 100 times, and all metrics recorded in Tables II and III were averaged across the test runs. The objective of Learn2Discover is to determine a label from the set {`fair`, `unfair`}. We evaluate Learn2Discover with respect to accuracy, precision, recall and F1-score.

Key results. We first consider the results of Learn2Discover run on each dataset with a stopping criterion of 250 annotations, in order to get a full picture of the human effort involved. As per Section VIII-B1, these tests were run with the least-confidence query strategy. The plots for each dataset are displayed in Figures 2 3 4 and 5 respectively. Each table is labelled at point (a, p), as per Section VI-F. These precision scores, along with the expected number of true positives for the given dataset, and relative human effort reduction, are detailed in Table II. Note that these values are evaluated with respect to the formulas for human effort defined in Section VI-F.

TABLE I. datasets

	COMPAS	Adult	German	Communities
Source	[3]	[41]	[42]	[43]
Size	60798	32561	1000	817
Attributes	28	15	21	101
Sensitive Attributes	2	2	3	3
True Fair:Unfair Ratio	55:45	67:33	22:78	86:14
Test: Train: Evaluate	1:6:3	1:6:3	35:35:30	35:35:30

TABLE II. Human effort required for 80% precision

	COMPAS	Adult	German	Communities
Annotations	10	60	30	80
True Positives	50495	26532	813	693
Human Effort Reduction	0.9998	0.8846	0.9631	0.8846

Next, in order to compare query strategies, the differences between evaluation metrics for least-confidence and entropy query strategies are displayed in Figure 6. This difference was evaluated on the COMPAS dataset across 250 annotations over 100 runs.

Finally, we compare each the effectiveness of Learn2Discover across each dataset with a stopping criterion of 45 annotations. This was determined from the mean number of annotations outlined in Table II. The metrics from these tests are shown in Table III.

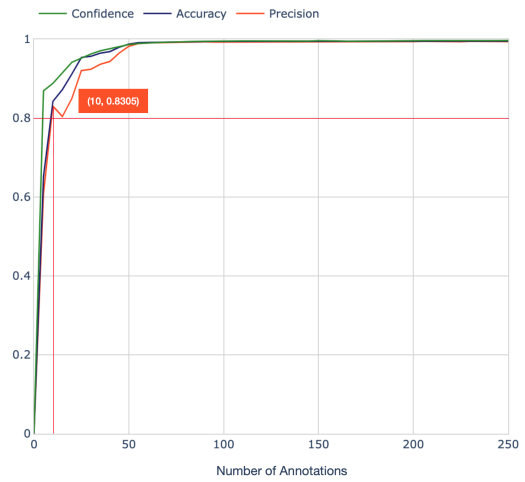

Fig. 2. The human effort in labelling COMPAS data

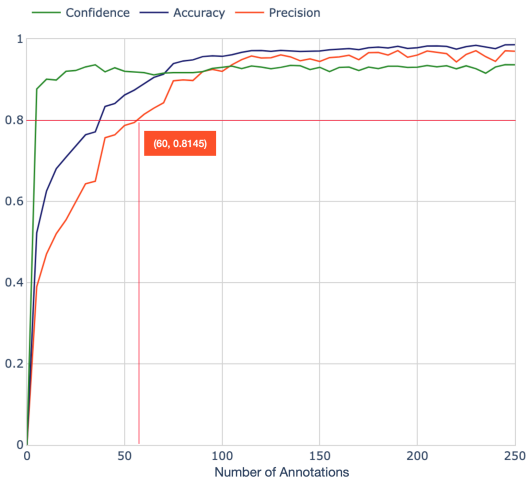

Fig. 3. The human effort in labelling Adult data

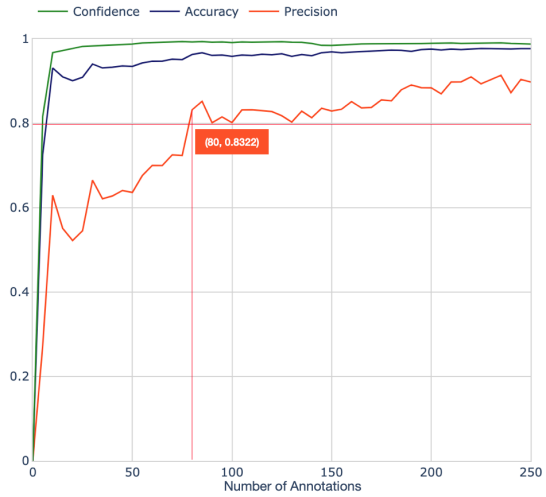

Fig. 5. The human effort in labelling Communities data

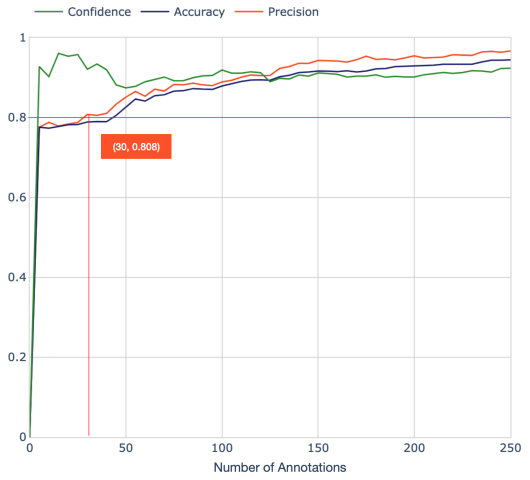

Fig. 4. The human effort in labelling German data

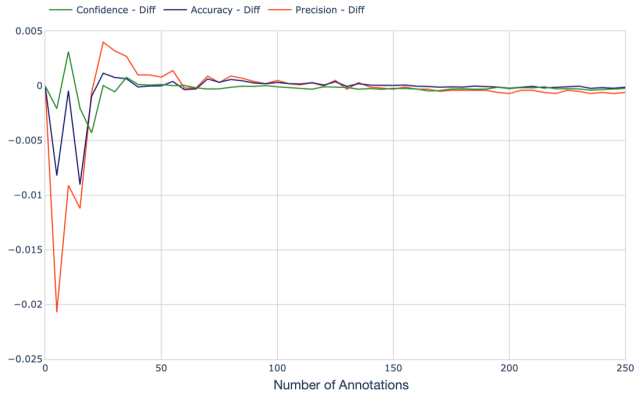

Fig. 6. Difference between Least Confidence and Entropy query strategies

VIII. Discussion

In evaluating Learn2Discover, we will discuss the results with respect to our research questions.

A. RQ1 - How much can Learn2Discover reduce the human effort involved in identifying unfairness?

For each dataset, based on the human effort reduction demonstrated in Table I, we can see that there is a significant reduction in human effort overall - ranging from approximately 85% for the Communities and Crime dataset, to over 99% in the Adult and COMPAS datasets.

TABLE III. The effectiveness of Learn2Discover

	COMPAS	Adult	German	Communities
Accuracy	0.9724	0.8481	0.7994	0.8985
Precision	0.9499	0.8445	0.8311	0.6444
Recall	0.9926	0.6439	0.9360	0.7038
F1-Score	0.9704	0.7217	0.8799	0.6719

The reason for the differences in results for these datasets will be outlined in response to *RQ2*. However, for the datasets evaluated, we expect an average of only 45 annotations to be required in order to achieve a precision of at least 80%. These results indicate that given a precision of 80%, we expect that Learn2Discover will yield a 96% overall reduction in human labelling effort compared to traditional manual labelling approaches.

B. RQ2 - How effective is Learn2Discover in identifying unfairness?

Before discussing the overall effectiveness of Learn2Discover, we must first discuss the two sub-questions under *RQ2*:

1) RQ2.1 - How does the choice of query strategy impact the active learning loop's performance?: Least confidence and entropy strategies are equivalent in the context of Learn2Discover.

Consider the formulations for least-confidence and entropy outlined in Section VI-D. Based on these formulations, we can see that entropy is directly derived from confidence. In

a system with two labels, such as Learn2Discover, the maximum entropy directly corresponds with minimum confidence.

In Figure 6, the y-scale has a range of of $[-0.025, 0.005]$, and the difference between the two query strategies tends towards 0 for each metric. Each iteration, the model takes a random sample of unlabelled data, which it then applies the query strategy to, followed by saving the newly labelled data. As the model's pool of labelled data increases, the variation between the two query strategies decreases; exposing the model to more data reduces the noise in its evaluation metrics. As such, we may conclude that the least confidence and entropy query strategies will yield approximately equivalent results, with negligible difference. However, entropy is a more computationally complex calculation, thus we conclude that least-confidence is the more suitable strategy.

2) RQ2.2 - How does the choice of stopping criteria impact the active learning loop's performance?: In comparing stopping criteria, three options were considered: a fixed number of annotations, a fixed number of iterations, and a fixed confidence level.

We first consider confidence as a stopping criterion: given Figures 2 3 4 and 5, the confidence fluctuates before approaching its resting value. For example, consider Figure 4, the graph for the German credit dataset. If the stopping criterion was set to a confidence level of 0.95, the learning loop would stop after only 15 annotations. However, the confidence level continues to drop below 0.95 after that point - in fact, it never reaches 0.95 again after 25 annotations.

Confidence is known to fluctuate in machine learning, as the model will usually be over-confident of its ability to correctly label data in early training iterations. However, as it is exposed to a wider variety of data, the confidence will tend towards a resting value. Therefore, confidence is not a stable stopping criterion for this approach.

The two remaining stopping criteria are number of iterations, and number of annotations. In Learn2Discover's active learning loop, a constant number of annotations are queried each iteration. As such, these two stopping criteria are functionally equivalent, and hence will yield the same results. However, in considering *RQ1*, being able to control the number of annotations provides extra utility in measuring and limiting human effort. Therefore, for the purpose of this research, number of annotations was selected as the stopping criterion, however using number of iterations will not affect performance.

3) Evaluation of effectiveness: Given the findings of *RQ2.1* and *RQ2.2*, we now seek to answer *RQ2*. In Table III, the accuracy, precision, recall, and F1-score for Learn2Discover have been outlined, under the configuration of: a least-confidence query strategy, and a stopping criterion of 45 annotations as per *RQ1*.

Before discussing these metrics, it is imperative to consider the potential causes of variation across datasets.

In Table I, the different characteristics of each dataset have been outlined. There are a wide range of dataset **sizes**, with COMPAS being the largest, at approximately 70 times the size of the smallest dataset, Communities and Crime. Recall that the final data in Table III has been averaged over 100 runs. For a larger dataset, there is a high likelihood that Learn2Discover will be exposed to a wide variety of training data. Conversely, for a smaller dataset, Learn2Discover is more likely to be exposed to repeated data across test runs. Therefore, the metrics evaluated on larger datasets are expected to be a better representation of the true effectiveness of Learn2Discover. Additionally, a larger dataset will increase Learn2Discover's range of exposure to attributes and corresponding labels.

Additionally, Table I outlines the **number of attributes** in each dataset. A dataset with many attributes is difficult to learn, due to there being more data to learn *from* per data instance. The Communities and Crime dataset has 101 attributes, whereas the other three datasets have between 15 and 28 attributes. In learning how to label a dataset with many attributes (such as Communities and Crime), Learn2Discover will be attempting to find patterns in each attribute with respect to its possible labels, leading to lower evaluation metrics.

Finally, the evaluation metrics are impacted by the **ratio of fair:unfair instances** of each dataset outlined in Table I. In a dataset with an approximately even split of fair and unfair data instances (such as COMPAS), we expect that the evaluation metrics will not be significantly impacted. However, datasets with a greater ratio of fair test cases (such as Communities and Crime) are expected to have lower precision, recall and F1 score than those with a greater ratio of unfair test cases (such as German Credit). This is due to there being a smaller pool of positive (i.e. unfair) test cases for training. Further, we expect recall to be more sensitive to this ratio than the precision; recall is calculated relative to the total number of truly unfair instances in the datasets, and as such, if Learn2Discover is exposed to a disproportionate number of unfair test cases, recall will be disproportionately affected.

We now consider the evaluation metrics outlined in Table III.

The COMPAS dataset has a large size and a small number of attributes, which both lead us to expect high values for the evaluation metrics. While the relatively balanced fair:unfair ratio is not expected to positively impact these metrics, any impact is likely to be offset by the large size of the dataset. As detailed in Table II, Learn2Discover was able to achieve the precision threshold after only 10 annotations. In Table III, we see that Learn2Discover achieved consistently high values across each metric for COMPAS. Therefore, we conclude that Learn2Discover behaved as expected and was able to successfully identify unfairness in the COMPAS dataset.

Similarly to the COMPAS dataset, the Adult dataset has a large size and few attributes. However, despite being large relative to German Credit and Communities and Crime, it is approximately half the size of COMPAS. Additionally, approximately $\frac{2}{3}$ of its data instances are labelled `fair`. Due to the higher proportion of `fair` instances, we expect

the precision, recall, and F1-score to be lower than those of COMPAS. Furthermore, we expect this proportion to cause an additional decrease to the recall. Finally, the smaller size of the Adult dataset also suggests that the accuracy should be lower than that of COMPAS. As detailed in Table II, Learn2Discover achieved the precision threshold after 60 annotations. In comparison to COMPAS, the Adult dataset yielded lower precision and significantly lower recall, in accordance with our expectations. Therefore, we conclude that Learn2Discover was able to successfully identify unfairness in the Adult dataset, but to a lesser degree compared to its results with COMPAS.

Unlike the COMPAS and Adult datasets, the German Credit dataset has a small size. However, it contains a small number of attributes, similarly to COMPAS and Adult. Conversely, approximately $\frac{4}{5}$ of the German Credit data instances are labelled `unfair`. Due to the smaller size of the German Credit dataset, we expect that the accuracy will be lowered. Additionally, the high proportion of `unfair` data instances suggests that precision will be increased, and recall will be significantly increased. These expectations are confirmed by the results shown in Table III. Furthermore, Learn2Discover achieved the precision threshold after 30 annotations, which is consistent with the increased precision. Therefore, we conclude that Learn2Discover behaved as expected and was able to successfully identify unfairness in the German Credit dataset, to a lesser degree than the COMPAS dataset and comparable to the Adult dataset.

Similarly to the German Credit dataset, the Communities and Crime dataset has a small size. However, unlike every other dataset, Communities and Crime has a very large number of attributes, which suggests that Learn2Discover might struggle to learn to identify unfairness for it. This also suggests that every evaluation metric might be lower compared to the other datasets. Communities and Crime has approximately $\frac{17}{20}$ data instances labelled `fair`, which we expect to decrease precision and recall similarly to the Adult dataset, but to a greater degree. Learn2Discover achieved the precision threshold after 80 annotations, which is consistent with our expectations based on the dataset's size and number of attributes. Considering its evaluation metrics in Table III, Communities and Crime has yielded a very high accuracy, but a comparatively lower precision and recall. This is consistent with our expectations, as the high proportion of `fair` test cases in conjunction with the high accuracy suggest that Learn2Discover has learned to identify `fair` test cases well. However, this proportion is also consistent with the lower precision and recall. Therefore, we conclude that Learn2Discover behaved as expected, and was considerably less successful at identifying unfairness in the Communities and Crime dataset.

IX. Limitations and Future Research

Given that the ultimate goal of Learn2Discover is to perform active learning using a human oracle to identify unfairness in software, a natural continuation of our work will be to extend the current implementation of Learn2Discover to directly satisfy this objective. In particular, this will entail developing some means of feeding unit tests for AI systems and their results into Learn2Discover, and developing an interface for humans to take the position of Learn2Discover's fairness oracle in real time. Additionally, due to humans being innately more nuanced, less predictable, and less consistent than computers, research will need to be conducted into verifying that Learn2Discover still produces meaningful results with a human oracle, compared to the current fully-automated oracle.

Furthermore, even without extending Learn2Discover to use human oracles, the currently used data generation approach can still be expanded upon. With just the four datasets considered in our research, we have found their results to be notably varied, so continuing our research with additional datasets may allow new conclusions to be drawn. Additionally, all generated datasets currently depend on a set of predefined, hard-coded sensitive attributes, so the data generation process could be expanded to introduce more variability to the sensitive attributes chosen, by either making their selection completely random, or by predefining a set of potentially sensitive attributes, and choosing a random non-empty subset of them as that run's sensitive attributes. In either case, increasing the number of possibilities for the selection of sensitive attributes may reveal different strengths and weaknesses of Learn2Discover, which can be iterated upon to improve its ability to identify unfairness.

As our research grants the assumption of fairness to the human oracle to learn domain-relevant biases, we make no attempt at proposing a normative procedure to mitigate human bias present in the training process. Additionally, a general approach to balancing accuracy-fairness tradeoffs is beyond the scope of this project, as the importance of one over the other is strictly context-dependent and varies with differing conceptions of justice [15].

Finally, Learn2Discover is part of a larger scale research initiative to develop means of identifying, understanding, and eliminating bias from AI systems through the use of a human-centric "human-in-the-loop" approach. Learn2Discover aims to solve the problem of identifying bias in AI, and will be followed by Learn2Explain to automatically explain unfairness in AI, and Learn2Improve to mitigate bias in AI.

X. Conclusion

In this paper we have proposed Learn2Discover, an active learning approach to fairness testing AI systems. The key contributions of Learn2Discover are a significant reduction in human effort in the labelling of test cases, as well as the integration of a human-in-the-loop. Learn2Discover provides a stepping stone for future developments in AI fairness testing, yielding an effective approach to identify cases of unfairness.

For reproducibility, Learn2Discover can be found at https://github.com/mjcalx/learn2discover.

References

[1] M. Bogen and A. Rieke, "Help wanted: an examination of hiring algorithms, equity, and bias," Upturn, Washington DC, USA, Tech. Rep., 2018.

[2] J. Angwin, J. Larson, S. Mattu, and L. Kirchner, "Machine bias," Propublica. Available at https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing (accessed Aug. 23, 2022), 2016.

[3] J. Larson, S. Mattu, L. Kirchner, and J. Angwin, "How we analyzed the COMPAS recidivism algorithm," Propublica. Available at https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm (accessed Aug. 29, 2022), 2016.

[4] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Comput. Surv.*, vol. 54, pp. 1–35, 2021.

[5] A. K. Manrai, B. H. Funke, H. L. Rehm, M. S. Olesen, B. A. Maron, P. Szolovits, D. M. Margulies, J. Loscalzo, and I. S. Kohane, "Genetic misdiagnoses and the potential for health disparities," *New England J. Med.*, vol. 375, pp. 655–665, 2016.

[6] W. E. Rice, "Race, gender, "redlining," and the discriminatory access to loans, credit, and insurance: an historical and empirical analysis of consumers who sued lenders and insurers in federal and state courts, 1950-1995," *The San Diego Law Review*, vol. 33, pp. 583–699, 1996.

[7] J. S. Dillbary and G. Edwards, "An empirical analysis of sexual orientation discrimination," *The University of Chicago Law Review*, vol. 86, no. 1, 2019.

[8] H. Sun and L. Gao, "Lending practices to same-sex borrowers," in *Proc. Nat. Acad. Sci.*, 2019, vol. 116, no. 19, pp. 9293–9302.

[9] T. Kamishima, S. Akaho, and J. Sakuma, "Fairness-aware learning through regularization approach," in *ICDM Workshops 2011, Proc. IEEE Int. Conf. Data Mining*, 2011, pp. 643–650.

[10] S. Galhotra, Y. Brun, and A. Meliou, "Fairness testing: testing software for discrimination," in *Proc. 2017 11th Joint Meeting Eur. Softw. Eng. Conf. and ACM SIGSOFT Symp. on Found. of Softw. Eng. (ESEC/FSE 2017)*, 2017.

[11] S. Udeshi, P. Arora, and S. Chattopadhyay, "Automated directed fairness testing," in *Proc. 33rd ACM/IEEE Int. Conf. Automated Softw. Eng. (ASE 2018)*, 2018.

[12] H. Zheng *et al.*, "NeuronFair: interpretable white-box fairness testing through biased neuron identification," in *2022 IEEE/ACM 44th Int. Conf. Softw. Eng. (ICSE)*, 2022, pp. 1519–1531.

[13] A. Agarwal, P. Lohia, S. Nagar, K. Dey, and D. Saha, "Automated test generation to detect individual discrimination in AI models," *CoRR*, vol. abs/1809.03260, 2018.

[14] D. Pessach and E. Shmueli, "A review on fairness in machine learning," *ACM Comput. Surv.*, vol. 55, no. 3, 2022.

[15] M. Srivastava, H. Heidari, and A. Krause, "Mathematical notions vs. human perception of fairness: a descriptive approach to fairness for machine learning," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery & Data Mining (KDD 2019)*, 2019.

[16] E. Mosqueira-Rey, E. Hernández-Pereira, D. Alonso-Ríos, J. Bobes-Bascarán, and Ángel Fernández-Leal, "Human-in-the-loop machine learning: a state of the art," 2022.

[17] T. Luo, K. Kramer, D. B. Goldgof, L. O. Hall, S. Samson, A. Remsen, and T. Hopkins, "Active learning to recognize multiple types of plankton," *Proc. 17th Int. Conf. on Pattern Recognit.*, vol. 3, pp. 478–481, 08 2004.

[18] K. D. Angeli, S. Gao, M. Alawad, H.-J. Yoon, N. Schaefferkoetter, X.-C. Wu, E. B. Durbin, J. Doherty, A. Stroup, L. Coyle, L. Penberthy, and G. Tourassi, "Deep active learning for classifying cancer pathology reports," *BMC Bioinf.*, vol. 22, no. 113, 03 2021.

[19] B. Settles, "From theories to queries: Active learning in practice," in *Active learning and experimental design workshop in conjunction with AISTATS 2010*. JMLR Workshop and Conference Proceedings, 2011, pp. 1–18.

[20] C. Longoni, A. Bonezzi, and C. K. Morewedge, "Resistance to medical artificial intelligence," 2019.

[21] M. K. Lee and S. Baykal, "Algorithmic mediation in group decisions: Fairness perceptions of algorithmically mediated vs. discussion-based social division," pp. 1035–1048, 2017.

[22] M. K. Lee and K. Rich, "Who is included in human perceptions of ai?: Trust and perceived fairness around healthcare ai and cultural mistrust," no. 138, pp. 1–14, 2021.

[23] E. T. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo, "The oracle problem in software testing: A survey," *IEEE Trans. Softw. Eng.*, vol. 41, no. 5, pp. 507–525, 2015.

[24] M. Harman, S. G. Kim, K. Lakhotia, P. McMinn, and S. Yoo, "Optimizing for the number of tests generated in search based test data generation with an application to the oracle cost problem," in *2010 3rd Int. Conf. Softw. Testing, Verification, and Validation Workshops*, 2010, pp. 182–191.

[25] S. Afshan and P. McMinn, "An investigation into qualitative human oracle costs," in *Psychol. of Program. Interest Group Annu. Workshop (PPIG 2011)*, 2011.

[26] P. McMinn, M. Stevenson, and M. Harman, "Reducing qualitative human oracle costs associated with automatically generated test data," in *Proc. 1st Int. Workshop Softw. Test Output Validation*, ser. STOV '10. Assoc. Comput. Machinery, 2010, pp. 1–4.

[27] R. A. Oliveira, U. Kanewala, and P. A. Nardi, "Chapter three - automated test oracles: state of the art, taxonomies, and trends," ser. Advances in Computers, A. Memon, Ed. Elsevier, 2014, vol. 95, pp. 113–199.

[28] Z. Chen, J. Zhang, M. Hort, F. Sarro, and M. Harman, "Fairness testing: a comprehensive survey and analysis of trends," *ArXiv*, vol. abs/2207.10223, 2022.

[29] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin, "Cost-effective active learning for deep image classification," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 27, no. 12, pp. 2591–2600, 2017.

[30] A. Perera, A. Aketu, C. Tantithamthavorn, J. Jiarpakdee, B. Turhan, L. Kuhn, and K. Walker, "Search-based fairness testing for regression-based machine learning systems," *Empirical Software Engineering*, vol. 27, no. 79, 2022.

[31] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[32] O. G. Rudenko, O. Bezsonov, and K. Oliinyk, "First-order optimization (training) algorithms in deep learning," in *COLINS*, 2020.

[33] K. Janocha and W. Czarnecki, "On loss functions for deep neural networks in classification," *Schedae Informaticae*, vol. 25, 02 2017.

[34] B. Ding, H. Qian, and J. Zhou, "Activation functions and their characteristics in deep neural networks," in *2018 Chinese Control And Decision Conference (CCDC)*, 2018, pp. 1836–1841.

[35] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *CoRR*, vol. abs/1710.05941, 2017.

[36] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[38] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[39] M. Buckland and F. Gey, "The relationship between recall and precision," *Journal of the American Society for Information Science*, vol. 45, pp. 12–19, 1994.

[40] J. M. Zhang, M. Harman, L. Ma, and Y. Liu, "Machine learning testing: survey, landscapes and horizons," *IEEE Trans. Softw. Eng.*, vol. 48, pp. 1,36, 2022.

[41] H. Hofmann, "Statlog (german credit data)," UCI Machine Learning Repository, 1994.

[42] "Adult," UCI Machine Learning Repository, 1996.

[43] "Communities and Crime," UCI Machine Learning Repository, 2009.