FairEM: Auditing Entity Matching Models for Fairness [Experiment, Analysis & Benchmark Papers]

Nima Shahbazi*
University of Illinois Chicago
nshahb3@uic.edu

Nikola Danevski*
University of Rochester
ndanevsk@u.rochester.edu

Fatemeh Nargesian
University of Rochester
fnargesian@rochester.edu

Abolfazl Asudeh
University of Illinois Chicago
asudeh@uic.edu

Divesh Srivastava
AT&T Chief Data Office
divesh@research.att.com

ABSTRACT

Entity matching is one of the first tasks in data integration pipelines. The new generation of entity matching techniques is heavily data- and learning-driven and potentially susceptible to injecting bias from data and pre-trained models into downstream tasks. We propose FairEM, an open-source library for auditing the fairness of learning-based entity matchers and providing explanations for the underlying reasons of unfairness. FairEM presents a suite of fairness measures and paradigms for evaluating the output of entity matchers. FairEM has a three-level architecture. The data layer allows a user to explore the space of subgroups in a hierarchical manner and choose subgroups of interest for auditing. In logic layer, the fairness of the output of a matcher is evaluated using various fairness measures. FairEM supports single and pairwise fairness for entity matching tasks. Finally, in the presentation layer, FairEM aggregates fairness results and provides insights on the overall fairness of the matcher as well as potential explanations for the unfairness of certain subgroups. We discuss interesting use-cases and findings from auditing the fairness of two state-of-the-art matchers, DeepMatcher and Ditto, on three benchmark data sets.

PVLDB Reference Format:
Nima Shahbazi, Nikola Danevski*, Fatemeh Nargesian, Abolfazl Asudeh, and Divesh Srivastava. FairEM: Auditing Entity Matching Models for Fairness [Experiment, Analysis & Benchmark Papers]. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at github.com/DataIntelligenceCrew/FairEM.

1 INTRODUCTION

In data cleaning and integration, entity matching is the task of identifying records from one or more data sources that refer to the same real-world entity. Entity matching technologies are key to solving a variety of complex tasks (e.g., building a knowledge graph and integrating heterogeneous data sets) and have been widely studied for over 50 years by the statistics, data management, NLP, and machine learning communities. Depending on the heterogeneity of data (e.g., the same information can be represented in structured fields or unstructured text) and quality of data (e.g., missing or erroneous values in the records), entity matching can be a difficult task. For these reasons, there is no single universally best matching algorithm [28] and effective entity matching remains a challenging task.

Fortunately, recent advancements in data technologies, especially in machine learning and AI, have provided promising solutions for complex entity matching tasks, significantly improving the effectiveness of these tasks. Modern entity matchers treat a matching task as a binary classifier on pairs of records [6, 29, 34]. The use of ML in the entity matching task has been shown to improve accuracy [6]. Modern entity matching systems successfully use deep learning for blocking [15] and leverage pre-trained transformer-based language models [29] and embedding vectors [34] for matching via fine-tuning. However, factors such as data quality and model choice may encode unintentional biases towards certain groups resulting in systematic disparate impact. That is, records from some groups may match at a significantly lower rate than records from other groups, with real-world consequences such as voter suppression or underestimating the prevalence of certain demographic groups. For instance, the use of pre-trained models gives rise to the possibility of propagating the known biases of pre-trained language models [30] into the matching outcome. Moreover, because learning-based entity matching systems often perform fine-tuning [29, 34], the distribution and sufficient representation (coverage) of groups of interests in the training data becomes crucial.

A systematic audit for unfairness is the first step to advance an entity matching algorithm towards being responsible. Assuming that no intentional bias is involved in creating a model for an application, data scientists should ensure that it is performing well enough for various groups or intersections of them before the model is operationalized.

We present FairEM, an evaluation platform for benchmarking and auditing the fairness of learning-based entity matchers. Our goal is to assist researchers and data practitioners in finding answers to the following questions.

- Is a matcher unfair/fair towards a group or sub-group of interest?
- What is the explanation for the unfairness of a matcher towards a group?

*Authors 1 and 2 equally contributed to this paper.

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

The main difference between the traditional notion of classification fairness with the notion of matching fairness lies in that in entity matching two corresponding records may belong to different groups. For example, consider the *iTunes-Amazon* data set with *genre* as its sensitive attribute. Given a pair to match, we may have one record belonging to *Country* and another record belonging to *Pop*. One audit question could be whether a matcher performs well for *Country* artists. Another question could be whether a matcher performs well for songs that belong to both *Country* and *Pop* genres. Similarly, can the model perform well when matching *Country* with *Pop* songs? To model this difference, we define *single* and *pairwise* fairness over groups and intersectional subgroups for matching tasks.

To audit the fairness of a matcher, we re-purpose the existing fairness measures defined for entity matching tasks. Some of these measures do not apply to pairwise fairness under certain circumstances where true label of an entity pair being a *match* implies the equality of their groups. For example, in the *DBLP-ACM* data set, two publications cannot be a true match if they are published in different venues. In such scenarios, measures such as True Positive Rate [7] do not apply, because when the groups of two entities to be matched are different, it is impossible for the ground-truth to be *match*, therefore number of true positives is always zero.

FairEM is a three-layer framework that provides a separation of functions across data layer, logic layer, and presentation layer. The data layer identifies groups and subgroups in sensitive attributes based on which the matcher should be evaluated. In particular, we differentiate between different types and number of sensitive attributes that impact the grouping, and provide an intersectional hierarchy for specifying the subgroups. The data layer also introduces a standard encoding that unifies grouping over different types and numbers of sensitive attributes and intersectional subgroups. Given a workload of a matcher's outcome, the logic layer evaluates the unfairness of the matcher with respect to the groups identified in the data layer and different fairness notions, both on single and pairwise fairness. To have a holistic view of a matcher, FairEM provides a way of combining fairness results on groups of interest using aggregate functions. The presentation layer of FairEM is responsible for aggregating fairness results, and to provide statistical testing analysis using multiple workloads of a matcher's output. Having identified unfairness towards a group, FairEM also enables further investigation of the sources of this unfairness by providing explanations based on 1) particular subgroups that are unfair, 2) the degree of difference (distance) between problematic pairs, and 3) synergies of other fairness measures with the measure of interest.

Using FairEM framework, we performed an extensive audit of two well-known entity matchers, Ditto [29] and DeepMatcher [34] on three benchmark data sets. The highlights of our observations are as follows.

- It is often the case that some subgroups are not sufficiently represented in test data to be audited, separately.
- Using pre-trained word embeddings and pre-trained transformer-based language models, in Amazon-iTunes music data set, caused an unfairness for genres such as `country`, where same artists have similar titles for different songs. That happened because the vector representation of the songs become very similar, causing the incorrect match. For example, Ditto mistakenly matched "`Tequila loves me`" and "`likes me`", both by `K. Chesney`. Besides, both models had lower accuracy (accuracy disparity) for one genre. For DeepMatcher, it was due to the fact that is with a higher probability predicted match pairs in that group as not match. Finally, other model details, such as (a) considering equal weights for different features in DeepMatcher and (b) putting too much weight on one feature (song title) in Ditto, caused other types on unfairness for different genres.
- Despite being well-presented in the training data, both matchers were unfair towards non-English languages, in our experiments on a *shoes* data set. Such unfairnesses can be explained by the unfairness of pre-trained language models used by the matchers.
- Our experiments on the DBLP-ACM data set, using multiple workloads, confirmed that both matchers were fair to all (DB) venues (present in the data set) with respect of all fairness measures. The reason was that the input data was not biased towards certain venues (and the training process and model details did not add unfairness), which resulted in fair matchers that performed equally good for different venues.

2 PRELIMINARY

2.1 Entity Matching

Given two sets of entities $A \in S_A$ and $B \in S_B$ from data sources S_A and S_B, the entity matching problem is to identify all correspondences between entities in $A \times B$ that correspond to the same real-world object. A correspondence $c = (e_i, e_j, s)$ interrelates two entities e_i and e_j with a confidence value $s \in [0, 1]$ that indicates the similarity of e_i and e_j or the confidence of a matcher about e_i and e_j referring to the same object. [28]. To decide whether the entity pair of $c = (e_i, e_j, s)$ is a *match* or *non-match*, matchers often apply a threshold on s [6, 53]. For auditing, we decouple the choice of a threshold from the outcome of the matching and consider the outcome of an entity matching task as pairs of matching and non-matching entities. Formally, we consider the following entity matching problem in this paper:

Definition 1 (Entity Matching Problem). *Consider two sets of entities $A \in S_A$ and $B \in S_B$ from data sources S_A and S_B. For every pair of entities $(e_i, e_j) \in A \times B$, let y_{ij} be the ground truth label indicating if e_i and e_j refer to the same object. Given all pairs $(e_i, e_j) \in A \times B$, the entity matching problem is to predict y_{ij} with a label h_{ij}. That is, h_{ij} refers to the decision of the matcher about the label of e_i and e_j (match or non-match).*

In a fairness-sensitive setting, entities are accompanied with sensitive attributes (e.g. `genre`, `language`, `race`, etc.). Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be the set of sensitive attributes, $dom(A_i)$ be the domain of A_i, and $\mathcal{G} = \{g_1, \ldots, g_m\}$ be the set of all groups of interest, i.e. $\mathcal{G} = \bigcup_{A_i \in \mathcal{A}} dom(A_i)$. The mapping $L(e_i)$ relates an entity to its associated groups $G_i \subseteq \mathcal{G}$. In other words, G_i is the group that e_i belongs to.

Given two sets of entities $A \in S_A$ and $B \in S_B$ from data sources S_A and S_B, and the set $[(e_i, e_j, G_i, G_j, h_{ij}, y_{ij})]_{\forall (e_i, e_j) \in A \times B}$, FairEM presents a framework for auditing the fairness of a matcher with respect to groups and combination of groups (subgroups), as we shall elaborate in the following.

2.2 Fairness

There is no singular definition for fairness. It all depends on the type of task we target to solve and the numerous kinds of bias that can exist in data. Since the focus of FairEM is mostly on auditing learning-based entity matching techniques, we provide a general overview of fairness definitions from the classification perspective. At a high level, fairness definitions can be viewed from three perspectives: group fairness, subgroup fairness, and individual fairness [7].

The most granular notion of fairness is individual fairness that requires similar outcomes for similar individuals [14]. The more popular perspectives of fairness for learning models, group/subgroup fairness, require similar treatment for different groups. A model satisfies some fairness constraints if the model has equal or similar performance (according to some fairness measure) on different subgroups. The focus of this paper is on group/subgroup fairness.

Most of the group fairness measures belong to one of the three categories of *Independence*, *Separation*, *Sufficiency*, and *Causation* [3, 7]. A model satisfies *independence* if its outcome is independent from the sensitive attributes. Measures such as *Statistical Parity* fall under this category. *Separation* is satisfied when the outcome of the model is independent from the sensitive attribute(s) conditioned on the target variable. The well-known measure in this category is *Equalized Odds* [23]. On the other hand, *Sufficiency* is satisfied if under the same model outcomes, sensitive attribute(s) and the true outcome are independent and it can be measured with *Predictive Parity*. *Causation* is however somehow different from the rest and focuses on the causal relationship between attributes for instance when an attribute A affects attribute B, which in turn affects attribute C. Domain experts are required to analyze a model in terms of Causality. Since, in FairEM, we only assume access to a matcher's decisions and true labels and not the whole data set, we choose to not to focus on Causal fairness in our audit.

Falling in between individual and group fairness, subgroup fairness (also known as intersectional fairness) metrics measure fairness (according to the above definitions) when groups are defined over intersection of values of multiple sensitive attributes (e.g. `white male`, `white female`, `black male`, and `black female`). In § 3, we describe how subgroups are defined and encoded in FairEM and in § 4.2, we describe how various group/subgroup fairness measures can be applied to entity matching tasks.

Regardless of the definition of fairness chosen to evaluate a model, there are challenges and domain-specific choices that need to be taken care of. Except for Equal Employment Opportunity Commission's 80% rule [12] that has been formalised as a bias measure, called Disparate Impact, there are no official guidelines for other measures. As a result, a model should be audited for groups of interest based on user-provided thresholds. Finding the subgroups for which a model is unfair is challenging. One reason is that a model can be fair with respect to any group individually but unfair towards a subgroup when fair groups are combined (e.g. a model that is fair both to `female` and `black`, can be unfair to `female black` subgroup). Identifying groups to which a model is discriminatory is particularly challenging due to the large space of possible subgroups. In § 3, we describe a hierarchical paradigm to exploring the space of all subgroups.

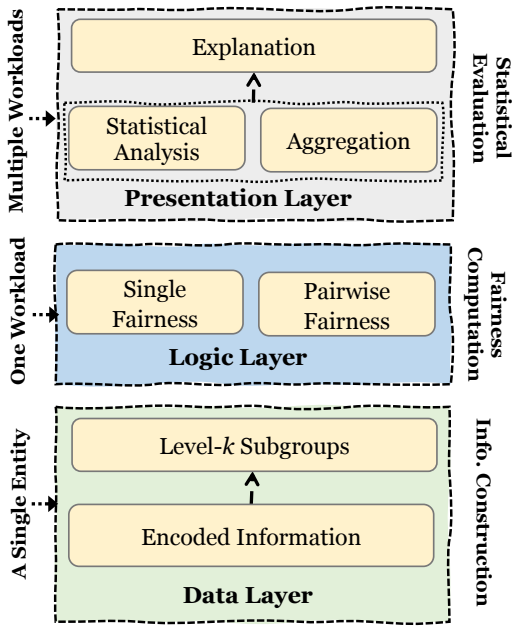

Figure 1: FairEM Architecture.

2.3 Fair Entity Matching

Fairness has so far been studied in many machine learning applications such as classification/regression/ranking tasks, recommendation systems, etc. Despite the inherent differences among these applications, they share a common characteristic which is the singularity of the object of reference. At a glance, entity matching may seem like a typical classification task, however, there are some distinctions as entity matching is a pairwise task by character, defined over a pair of entities. Consequently, evaluating learning-based entity matchers is different from classification models as the performance of the model over groups from both entities need to be considered. For this very reasons, we also need to repurpose the group fairness measures for the entity matching task.

2.4 FairEM Architecture

FairEM is a three-layer framework that given entity matching test data enables human-in-the-loop exploration of groups of interest in a matching test data set, audit of the fairness of groups, and explain group unfairness. Figure 1 shows the architecture of FairEM. The architecture provides a separation of functions in the three layers: *data*, *logic* and *presentation* layers. The three-layer architecture facilitates upgrade and maintenance of each layer independent of others. While the initial input to this architecture, particularly to the data layer, is an entity matching test data set containing pairs of $(e_i, e_j, G_i, G_j, m, y)$ in a standard format, the user can directly interact with each layer by specifying parameters and asking questions.

The main goal of data layer is to identify and encode groups of interest. We allow users to explore groups and subgroups through a hierarchical data structure. Given the type and number of sensitive attributes and cardinality of each, the data layer presents a standardized encoding of sensitive attributes that is capable of handling different possible cases. The logic layer aims at evaluating user-specified fairness measures over the groups identified in data

layer. Finally, in the presentation layer, the evaluation results from logic layer are aggregated, statistically analyzed for the user. In addition to fairness analysis, the presentation layer enables users to identify various types of explanations for the sources of unfairness. In the following sections, we discuss each layer in detail.

3 DATA LAYER: GROUP ENCODING

3.1 Group Selection

The first step in auditing an entity matcher for fairness is identifying meaningful groups/subgroups in sensitive attributes based on which the matcher should be audited. Sensitive attributes are attributes for which a matcher is likely to exhibit bias. These attributes are identified by the user and their values are associated to entities prior to being passed to the data layer. More concretely, an input file from matcher $\mathcal{M}$ to the data layer includes entity ids, the value of each entity for sensitive attributes, the decisions of $\mathcal{M}$ as well as true labels for the entity pairs. Note that FairEM requires access to the sensitive attributes and *ground truth* values for entities.

Depending on the type, cardinality and number of sensitive attributes multiple fairness cases may happen:

- *Single attribute with binary values:* In this case, fairness is studied on a single sensitive attribute. Each entity belongs to one of the two groups. e.g. `gender={male, female}`.
- *Single attribute with multiple (exclusive) values:* In this case, fairness is studied on a single sensitive attribute. Each entity belongs to one of the multiple demographic groups. e.g. `gender={male, female, transgender, non-binary, other}`.
- *Single setwise attribute:* In this case, fairness is studied on a single sensitive attribute. Each entity can belong to a subset of the universe of possible values of an attribute. For example, assuming `genre={Pop, Rock, Jazz}`, an entity e can have multiple genres: `genre(e)={Pop, Rock}`. Notice that in this case, the identified groups can have overlapping values, e.g. for entities e_1 and e_2, `genre`(e_1)`={Pop, Rock}`, `genre`(e_2)`={Pop, Jazz}`.
- *Multiple attributes:* In this case, fairness is studied on the intersection of multiple sensitive attributes. The values could be either one or a combination of the three cases discussed before. An example includes the groups defined on intersection of (single setwise attribute) `genre` and (binary attribute) `gender`: `{male-Pop, male-Rock, male-Jazz, female-Pop, female-Rock, female-Jazz, male-Pop-Rock, male-Pop-Jazz, male-Rock-Jazz, female-Pop-Rock, female-Pop-Jazz, female-Rock-Jazz, male-Pop-Rock-Jazz, female-Pop-Rock-Jazz}`.

For single attribute with binary or multiple values, the space of groups if interest is the number of values in the domain of the corresponding attribute. For example, a matcher is audited for `male`, `female`, ... separately. Single setwise attribute and multiple attributes allows us to define intersectional subgroup with various combinations of groups from different sensitive attributes. We represent the space of possible subgroups in a hierarchical data structure, where the first layer includes groups of all attributes. Level k includes a set of k non-overlapping groups created by combining groups from k different attributes with binary or multiple values, or $k-1$ groups from a setwise attribute with one group from a binary or multi-value attribute, and so on.

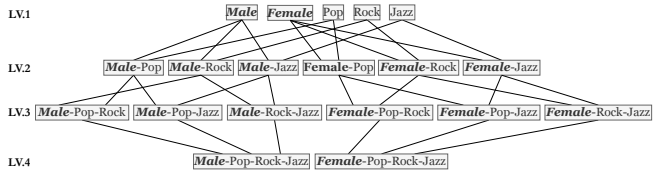

Figure 2: Intersectional subgroup hierarchy for single setwise and multiple attributes

Example 1: Figure 2 shows the intersectional subgroup hierarchy of sensitive attributes `gender` and `genre` for a data set matching songs of different artists. Note that `gender` is a binary attribute and `genre` is a setwise attribute. The level-2 of this hierarchy includes all combinations of groups from `gender` and `genre` in level-1. Level-3 enumerates 2-combinations of the domain of `genre` with groups from `gender`. □

Note that a subgroup hierarchy represents the space of groups and does not mean a data set must or do contain all these groups. In addition to enabling fairness audit on a particular group selected by a user, FairEM allows batch auditing subgroups of each level. That is, as we will show in our experiments, the fairness of a matcher is evaluated and compared with respect to all subgroups of a particular level selected by a user.

Next, we describe how FairEM encodes the input data to interact with other layers of the framework.

3.2 Group Encoding

Having created the space of groups, the next challenge is to develop a standard notation to unify all of the aforementioned attribute-value cases. To do so, we propose an encoding that summarizes sub-groups and use this encoding to represent individual entities and entity pairs.

Given a set of sensitive attributes $\mathcal{A} = \{A_1, \ldots, A_n\}$ and value domains $dom(A_i)$ for attributes A_i, $\mathcal{G} = \{g_1, \ldots, g_m\}$ denotes the set of all level-1 groups, i.e. $\mathcal{G} = \bigcup_{A_i \in \mathcal{A}} dom(A_i)$. We represent a subgroup s of level k (k-combination) consisting of groups $s = \{g_1, \ldots, g_k\}$, with a binary encoding $s = \langle a_1, \ldots, a_m \rangle$, where $m = |dom(A_1)| \times \ldots |dom(A_i)|)$ and a_i is one if $g_i \in s$ and is zero otherwise. Note that for a k-combination subgroup, exactly k entries of s get the value one. We represent an entity e associated with groups $G \subseteq \mathcal{G}$ with a binary encoding $\langle b_1, \ldots, b_m \rangle$, where $m = |dom(A_1)| \times \ldots |dom(A_i)|)$ and b_i is one if $g_i \in G$ and is zero otherwise.

Example 2: Consider attributes `genre` and `gender` of Figure 2. Assuming a lexicographical order on all groups, the encoding of entity e with associated groups G={`Female`, `Pop`, `Rock`} is $\langle 1, 0, 0, 1, 1 \rangle$. The encoding of a level-2 subgroup s={`Female`, `Pop`} is $\langle 1, 0, 0, 1, 0 \rangle$. □

Entity encoding is the output of the data layer and will be passed as an input to the logic layer, where fairness of a matcher is investigated with respect to a subgroup. An entity e with groups G belongs to subgroup s if $s \subseteq G$. Given an entity encoding $e = \langle b_1, \ldots, b_m \rangle$ and a subgroup encoding $s = \langle a_1, \ldots, a_m \rangle$, we say e belongs to

subgroup s if s *AND* $e == s$, i.e. the entity belongs to every group that define the subgroup s.

Example 3: Continuing with Example 1, a pop-rock entity e from a female singer belongs to subgroup `Female-Pop` and is counted in the audit of this subgroup. Following the same convention, the encoding of an entity pair e_i, e_j is the concatenation of the encodings of e_i and e_j. □

4 LOGIC LAYER: FAIRNESS COMPUTATION

In the data layer, we are concerned with the selection and proper representation of groups. Logic layer is where the actual evaluation of the output of an entity matching task takes place. The input to the logic layer is a workload w of n tuples each having a correspondence $t = (e_i, e_j, h, y)$, where h is a binary variable indicating the result of entity matching (*match* or *non-match*) for entities with encodings e_i and e_j, and y is a binary variable indicating the ground-truth for matching. A workload is defined as a test set of tuples for evaluating an entity matcher.

Given the pairwise nature of entity matching tasks, there are two ways to audit entity matchers:

- *Single:* In the single fairness, the performance of a matcher is evaluated for one subgroup s against either entity in a pair. Given a correspondence (e_i, e_j, h, y) and a subgroup s of interest, FairEM considers the correspondence legitimate, if either e_i or e_j belong to subgroups s. Note this can be easily verified using our binary encodings.
- *Pairwise:* In the pairwise fairness, the performance of a matcher is evaluated for a pair of subgroups s, s' against both entities in a pair. Given a correspondence (e_i, e_j, h, y) and a pair of sub-groups s, s' of interest, FairEM considers the correspondence legitimate, if either e_i or e_j belong to subgroups c. if e_i belongs to s and e_j belongs to s', or vice versa. From an encoding perspective, FairEM concatenates the encodings of subgroups s and s' into a vector c and the encodings of e_i and e_j into a vector e and validates vector e belongs to c with both directions of $\langle s, s' \rangle$ and $\langle s', s \rangle$.

In single and pairwise definitions, we consider the entity matching task to be symmetric. We remark that these definitions can be extended to ordered single and ordered pairwise fairness where the subgroups are defined on left or right entities. In this paper, we focus on non-directional single and pairwise fairness.

Given a subgroup of interest, the logic layer summarizes the workload w into a confusion matrix, which is later used in computing various fairness measures.

4.1 Creating Confusion Matrix

When auditing with single and pairwise fainress, each correspondence (e_i, e_j, m, y) corresponds either to True Positives (TPs), False Positives (FPs), False Negatives (FNs), or True Negatives (TNs). Note that the result is counted both for the group(s) of e_i and the group(s) of e_j. This is unlike regular classification where every row corresponds to one entity and hence is counted once. It is worth describing how a confusion matrix is created for an entity matching task through an example.

Example 4: Consider an input data set from matcher $\mathcal{M}$ shown in Table 3, where columns id_1 and id_2 contain entity encodings, column h is the output decision of $\mathcal{M}$, and column y is the ground-truth. Comparing columns h and y, we add and populate column *Result* for each entity pair. Note that this is independent of the audit subgroup of interest. Consider the simple case of having two groups $\mathcal{G} = \{g_1, g_2\}$. An entity e_i in id_1 would be represented with an encoding of size 2. Suppose we would like to audit single fairness for groups g_1 and g_2. We describe how the confusion matrices of these groups are created. Consider the first row in Table 3a and it happens to be an FP. Since e_1 and e_2 both belong to subgroup g_1, the value two will be added to the count of FPs in the confusion matrix of g_1. However, in the second row which happens to be a TN, e_3 belongs to g_2 while e_4 belongs to g_1. Thereby, we will add one to both TN values of the confusion matrix corresponding to subgroup g_1 and g_2. We repeat the same procedure for rows three and four. The completed confusion matrices are shown in Figures 3b and 3c. □

id_1	id_2	$enc_1\langle g_1, g_2\rangle$	$enc_2\langle g_1, g_2\rangle$	h	y	Result
e_1	e_2	$\langle 1, 0\rangle$	$\langle 1, 0\rangle$	'M'	'N'	FP
e_3	e_4	$\langle 0, 1\rangle$	$\langle 1, 0\rangle$	'N'	'N'	TN
e_1	e_4	$\langle 1, 0\rangle$	$\langle 1, 0\rangle$	'M'	'M'	TP
e_2	e_3	$\langle 1, 0\rangle$	$\langle 0, 1\rangle$	'N'	'M'	FN

(a)

Predicted \ Actual	y='M'	y='N'
h='M'	TP=2	FP=2
h='N'	FN=1	TN=1

(b)

Predicted \ Actual	y='M'	y='N'
h='M'	TP=0	FP=0
h='N'	FN=1	TN=1

(c)

Figure 3: (a) Matching Results (b) Confusion Matrix of g_1 (c) Confusion Matrix of g_2.

4.2 Entity Matching Fairness Measures

As explained in Section 2.2, depending on the context of an entity matching task at hand, proper fairness measures should be employed. More precisely, the adopted fairness measures depend on the importance of TPs, FPs, FNs, and TNs in the problem context and how forgiving we can be towards each.

We note that some measures cannot be applied in pairwise fairness scenarios where conceptually the equality of groups restricts matching results. In some scenarios two entities with different groups can never be considered as a *match* in the ground-truth. For instance, in a matching task defined between *DBLP* and *ACM* publications, two entities with different venues (after standardization) and years are never a true *match*. More concretely, when pairwise fairness is evaluated on subgroups with non-overlapping groups, TPs and FNs are always zero, therefore, fairness measures that rely on TPs and FNs become inapplicable.

In what follows, we present a suite of fairness measures, based on notions of fairness in classification [7] and repurposed for entity matching, for auditing an entity matcher $\mathcal{M}$ with respect to a set $\mathcal{G}$ of k-combination subgroups. Let $h(e, e')$ be the output of matcher

$\mathcal{M}$ (*match* ('M') or *non-match* ('N')) and y be the ground-truth on entities e and e'.

- **Accuracy Parity (AP)** equalizes matchers's accuracy across different subgroups:

$$\forall g_i \in \mathcal{G}, Pr(h(e,e') = y | g_i) \simeq Pr(h(e,e') = y) \tag{1}$$

  Accuracy parity is a useful measure in the contexts where the impact of FPs and FNs are similar. This measure extends to both *single* and *pairwise* fairness definitions. Alternatively, instead of accuracy, one can consider the **misclassification rate parity**:

$$\forall g_i \in \mathcal{G}, Pr(h(e,e') \neq y \mid g_i) \simeq Pr(h(e,e') \neq y) \tag{2}$$

- **Statistical Parity (SP)** makes the independence assumption of the matcher from groups ($h \perp \mathcal{G}$). Under statistical parity [14], the probability of a matcher outcome is equal or similar across different groups:

$$\forall g_i \in \mathcal{G}, Pr(h(e,e') = \text{'}M\text{'} \mid g_i) \simeq Pr(h(e,e') = \text{'}M\text{'}) \tag{3}$$

  Statistical parity is not an applicable fairness measure in deduplication tasks using entity matching since equal probability of getting a match given a subgroup may not be meaningful, however, this measure is useful for entity matching in table joins. Furthermore, this measure extends to both *single* and *pairwise* fairness definitions.
- **True Positive Rate Parity (TPRP)** also known as *Equal Opportunity* or *Sensitivity* is a practical measure when predicting the positive outcome correctly is crucial and FPs are not costly:

$$\forall g_i \in \mathcal{G}, Pr(h(e,e') = \text{'}M\text{'} | g_i, y = \text{'}M\text{'}) \simeq Pr(h(e,e') = \text{'}M\text{'} | y = \text{'}M\text{'}) \tag{4}$$

  This measure is only meaningful for *single* fairness and is not definable in *pairwise* fairness. The reason is that when the groups of two entities to be matched are different, it is impossible for the ground-truth to be *match*, therefore number of TPs is always zero.
- **False Positive Rate Parity (FPRP)** is a useful measure in contexts that minimizing costly FPs matter most:

$$\forall g_i \in \mathcal{G}, Pr(h(e,e') = \text{'}M\text{'} | g_i, y = \text{'}N\text{'}) \simeq Pr(h(e,e') = \text{'}M\text{'} | y = \text{'}N\text{'}) \tag{5}$$

  This measure extends to both *single* and *pairwise* fairness definitions.
- **False Negative Rate Parity (FNRP)** is a useful measure in contexts that FNs are costly and the overweighing source of error:

$$\forall g_i \in \mathcal{G}, Pr(h(e,e') = \text{'}N\text{'} | g_i, y = \text{'}M\text{'}) \simeq Pr(h(e,e') = \text{'}N\text{'} | y = \text{'}M\text{'}) \tag{6}$$

  For the same reasons, this measure is also only meaningful for *single* fairness and does not extend to *pairwise* fairness. Entities with different groups can not have a *match* ground-truth.
- **True Negative Rate Parity (TNRP)** also known as *specificity* measures model's ability to correctly identify negative results:

$$\forall g_i \in \mathcal{G}, Pr(h(e,e') = \text{'}N\text{'} | g_i, y = \text{'}N\text{'}) \simeq Pr(h(e,e') = \text{'}N\text{'} | y = \text{'}N\text{'}) \tag{7}$$

  This measure extends to both *single* and *pairwise* fairness definitions.

Fairness Measure	Single	Pairwise
Accuracy Parity	✓	✓
Statistical Parity	✓	✓
True Positive Rate Parity	✓	—*
False Positive Rate Parity	✓	✓
False Negative Rate Parity	✓	—*
True Negative Rate Parity	✓	✓
Equalized Odds	✓	—*
Positive Predictive Value Parity	✓	—*
Negative Predictive Value Parity	✓	—*
False Discovery Rate Parity	✓	—*
False Omission Rate Parity	✓	—*

Figure 4: Entity matching fairness measure extendability in *single* and *pairwise* fairness. * Not extendable to domains where a true match requires the overlap of groups of entities, otherwise extendable.

- **Equalized Odds (EO)** also known as *Positive Rate Parity* is practical in contexts that correctly predicting positive outcomes and minimizing costly FPs are both of high importance:

$$\forall g_i \in \mathcal{G}, Pr(h(e,e') = \text{'}M\text{'} | g_i, y = \text{'}M\text{'}) \simeq Pr(h(e,e') = \text{'}M\text{'} | y = \text{'}M\text{'})$$
$$Pr(h(e,e') = \text{'}M\text{'} | g_i, y = \text{'}N\text{'}) \simeq Pr(h(e,e') = \text{'}M\text{'} | y = \text{'}N\text{'}) \tag{8}$$

  This measure is also only meaningful for *single* fairness and does not extend to *pairwise* fairness.
- **Positive Predictive Value Parity (PPVP)** guarantees equal chance of success given positive prediction for all subgroups:

$$\forall g_i \in \mathcal{G}, Pr(y = \text{'}M\text{'} | h(e,e') = \text{'}M\text{'}, g_i) \simeq Pr(y = \text{'}M\text{'} | h(e,e') = \text{'}M\text{'}) \tag{9}$$

  This measure is also only meaningful for *single* fairness and does not extend to *pairwise* fairness.
- **Negative Predictive Value Parity (NPVP)** ensures equal chance of success given negative prediction for all subgroups:

$$\forall g_i \in \mathcal{G}, Pr(y = \text{'}N\text{'} | h(e,e') = \text{'}N\text{'}, g_i) \simeq Pr(y = \text{'}N\text{'} | h(e,e') = \text{'}N\text{'}) \tag{10}$$

  This measure is also only meaningful for *single* fairness and does not extend to *pairwise* fairness.
- **False Discovery Rate Parity (FDRP)** makes the independence assumption of true match/non-match decision from subgroups, conditional on the *match* decision ($y \perp \mathcal{G} |\ h(e,e') = \text{'}M\text{'}$).

$$\forall g_i \in \mathcal{G}, Pr(y = \text{'}N\text{'} | g_i, h(e,e') = \text{'}M\text{'}) \simeq Pr(y = \text{'}N\text{'} | h(e,e') = \text{'}M\text{'}) \tag{11}$$

  This measure is also only meaningful for *single* fairness and does not extend to *pairwise* fairness.
- **False Omission Rate Parity (FORP)** makes the independence assumption of true match/non-match decision from subgroups, conditional on the *non-match* decision ($y \perp \mathcal{G} |\ h = 0$).

$$\forall g_i \in \mathcal{G}, Pr(y = \text{'}M\text{'} | g_i, h(e,e') = \text{'}N\text{'}) \simeq Pr(y = \text{'}M\text{'} | h(e,e') = \text{'}N\text{'}) \tag{12}$$

  This measure is also only meaningful for *single* fairness and does not extend to *pairwise* fairness.

A summary of entity matching fairness measures and their definability for *single* and *pairwise* fairness definitions is shown in Figure 4.

4.3 Entity Matching Fairness Disparity (Unfairness)

Consider a fairness notion and a subgroup $g_i \in \mathcal{G}$. In a perfect situation, the matcher should satisfy the parity (equality) between two probabilities in the following form:

$$\forall g_i \in \mathcal{G}, Pr(\alpha \mid \beta, g_i) = Pr(\alpha \mid \beta) \tag{13}$$

where α and β are specified by the fairness measure. For example, for Positive Predictive Parity, α is $y =$ 'M' and β is $h(e, e') =$ 'M'.

On the other hand, due to the trade-offs [26] between different fairness notions and the impossibilities theorems [10], it is often not possible to satisfy complete parity on all fairness measures. As a result, considering a threshold value (e.g. the 20% rule [18] suggests the threshold as 0.2), the objective is to make sure that *disparity* (as known as *unfairness*) is less than the threshold. Given a fairness notion and a subgroup $g_i \in \mathcal{G}$, the disparity can be computed using subtraction [8], as following:

$$F^{(s)}_{\alpha,\beta}(g_i) = \max\Big(0\,,\ Pr(\alpha \mid \beta) - Pr(\alpha \mid \beta, g_i)\Big) \tag{14}$$

For example, for accuracy parity (α is $h(e, e') = y$ and β is null) the disparity can be computed as

$$F^{(s)}_{\text{AP}}(g_i) = \max\Big(0\,,\ Pr(h(e, e') = y) - Pr(h(e, e') = y \mid g_i)\Big)$$

Note that if the accuracy for the subgroup g_i is higher than the average accuracy of the model, it is not considered as unfairness. Also, note that Equation 14 considers the higher the probability the better. Depending on fairness measures (and application), the direction may be as the lower the probability the better. For example, for FNRP, lower probability of false negative is preferred. For such cases, one should consider $Pr(h(e, e') = y \mid g_i) - Pr(h(e, e') = y)$ when computing disparity. As a result, for false negative rate (α is $h(e, e') =$ 'N' and β is $y =$ 'M') the disparity can be computed as

$$\begin{aligned} F^{(s)}_{\text{FNRP}}(g_i) = \max\Big(0\,, Pr(h(e, e') = 0 \mid y = \text{'}M\text{'}, g_i) \\ - Pr(h(e, e') = 0 \mid y = \text{'}M\text{'})\Big) \end{aligned} \tag{15}$$

Alternatively, given a fairness notion and a subgroup $g_i \in \mathcal{G}$, the disparity can be computed using division [18, 18] , as following:

$$F^{(d)}_{\alpha,\beta}(g_i) = \max\Big(0\,, 1 - \frac{Pr(\alpha \mid \beta, g_i)}{Pr(\alpha \mid \beta)}\Big) \tag{16}$$

Similar to Equation 14, Equation 16 also considers the higher probabilities the better. For the cases (such as FNRP or FDRP) where the lower probabilities are better, one should swap the nominator and the denominator in the equation. Therefore, for false discovery rate (α is $y = 0$ and β is $h(x) = 1$) the disparity can be computed as

$$F^{(d)}_{\text{FDRP}}(g_i) = \max\Big(0\,, 1 - \frac{Pr(y = \text{'}N\text{'} \mid h(e, e') = \text{'}M\text{'})}{Pr(y = \text{'}N\text{'} \mid h(x) = \text{'}M\text{'}, g_i)}\Big)$$

Our proposal in this paper is agnostic to the choice of operation for computing the disparities. Still, in our experiments, without any preference, we use subtraction for computing the disparities.

5 PRESENTATION LAYER

5.1 Statistical Evaluation

The presentation layer analyzes the results generated by the logic layer from evaluating fairness measures on subgroups of interest. More concretely, the input to the presentation layer is the disparity values of each subgroup in case of single fairness or each subgroup pair in case of pairwise fairness, for all applicable measures. Depending on user preferences and available data, the results are analyzed in two settings.

Single-workload Analysis: In this setting, the disparity results are available for one test data set. In a group analysis perspective, the goal is to identify which subgroups are unfair with respect to which measure. This can be done by comparing the disparity of each subgroup for each measure with a corresponding user-provided threshold. In a measure analysis perspective, the goal is to aggregate subgroup disparities to make a final conclusion about the fairness of a matcher based on a measure. Focusing on a measure, a user may choose to aggregate disparity results of all subgroups using functions such as `MAX`, `MIN`, `AVG`, and `MAX` minus `MIN`. Comparing the aggregated value with a disparity threshold determines whether a matcher is overall unfair with respect to a particular measure.

Multiple-workload Analysis: We consider the scenario where multiple instances of test data (workloads) are available from a matcher. For example, different test data may become available at different times or different samples from the underlying test data distribution may be available.

In this case, the logic layer evaluates workloads separately. For k workloads, the input to the presentation layer is a population of k disparity values for each subgroup and measure combination. A population for subgroup s and measure m includes the disparity of s in every workload with respect to m. To assess the fairness of a matcher $\mathcal{M}$ on subgroup c using measure m and k workloads, we employ the standard *hypothesis testing* [37]. For a subgroup s and measure m, the fairness hypothesis testing considers the *null* hypothesis that the matcher is fair on s and the *alternative* hypothesis that the matcher is unfair on s. For more than one workload, FairEM chooses the appropriate z-test statistics. Next, the test statistics and corresponding p-value are computed as the probability of getting the observed test statistic or something more extreme when the null hypothesis is true. Finally, given a significance level α, the null hypothesis in favor of the alternative is rejected if $\alpha \leq p$-value, or not if $\alpha > p$-value.

If a user chooses to perform multiple-workload analysis on a single provided test data set, FairEM generates k workloads by random sampling with replacement from the data set. The goal of this analysis would be to verify whether an unfair subgroup happened by chance or is indeed repeatable.

5.2 Explainability

After a matcher is audited for fairness and groups subject to unfairness are identified, FairEM reassures to offer additional insights explaining the unfairness towards a group. FairEM approaches the explainability of group unfairness in three ways, discussed in the following sections. The explanations provided by FairEM falls under the category of *Local Model-agnostic Methods* [33], where given an unfairness measure and a group towards which the model

has been unfair, the goal is to provide (local) explanations for the queried (measure, group).

The presentation layer determines whether a matcher is unfair overall or for a subgroup, with respect to a measure. To allow users to explore potential explanations for unfairness, FairEM provides three perspectives.

5.2.1 Subgroup-based Explanation. A matcher may be unfair on a subgroup s because it performs poorly on more granular subgroups of s. Navigating the subgroup hierarchy of a matcher downward from an unfair subgroup node and considering matcher's performance on descendant nodes allows us to identify the subgroups of s that may be the source of unfairness. Given a k-level subgroup s, FairEM offers the ability to investigate the unfairness for subgroups located in m levels deeper than s in the subgroup hierarchy, such that $1 \leq m \leq k - depth(g)$. Assuming sufficient data exists for s and its granular subgroup in the data set, disparity analysis of these subgroups over various measures allows the user to gain more insights on the unfairness of s.

Example 5: Consider the subgroup hierarchy of Figure 2 for a data set with two sensitive attributes `genre` and `gender`. Suppose a matcher is unfair based on accuracy disparity towards `Female` group, which is a level-1 subgroup. FairEM evaluates all the level-2 subgroups that have `Female` as their parent i.e. `Female-Pop`, `Female-Rock`, etc. Assuming that the observations show that the matcher is fair towards `Female-Pop`, `Female-Rock` and unfair towards `Female-Jazz`, one can explain the reasons for unfairness towards `Female` subgroup due to matcher's bad performance towards `Female-Jazz` subgroup. □

5.2.2 Distance-based Explanation. Subgroup-based explanations allow us to pinpoint the fine-grained subgroups that might have caused unfairness for a subgroup of interest. Distance-based explanations provide an abstract view over theses subgroups. In distance-based perspective, FairEM explores how a matcher performs on entity pairs belonging to subgroup s that also belong to various other subgroups. Let us define distance for between entities e_i and e_j in tuple $(e_i, e_j, G_i, G_j, m, y)$. Suppose both entities belong to subgroup s, i.e. $s \subseteq G_i$ and $s \subseteq G_j$. The distance of $d(e_i, e_j) = |G_i \setminus G_j| + |G_j \setminus G_i|$. We would like to know how the distance of group associations of entities impact the fairness of a model. To explain the unfairness for s, for single (pairwise) fairness FairEM groups all tuples that have s in at least one entity (in both entities) and evaluates fairness measures for each distance group. The goal is to specify if the performance of the matcher on different distances within a group g, is similar to the performance of the matcher on different distances over all groups. One motivation for using distance-based explanations is that often a data set may not have sufficient tuples from highly fine-grained subgroups and distance-based grouping allows us to make statistically significant conclusions.

We note that the distance-based explanation does not extend for TPRP and FNRP. The reason is that when e_i and e_j are match, i.e., correspond to the same real-world object, $G_i = G_j$. As a result, the distance between groups for true positive and false negative cases is always zero.

5.2.3 Measure-based Explanation. The measure-based explanation describes the unfairness of a matcher in terms of other measures. This is a common practice in analyzing model performance holistically. This explanation is useful in particular for explaining accuracy disparity. Considering the confusion matrix (e.g. Figure 3-b), low accuracy of the model for a specific group can be due to the bad performance on one of the matrix cells such as true positive. In such a situation, unfairness on the given cell (e.g. TPRP) explains the unfairness with respect to AP. In § 7, for example, we show how the Accuracy disparity of a matcher can be explained by its high False Positive Rate disparity.

6 IMPLEMENTATION

In this section, we discuss the implementation details of FairEM. The class diagram for FairEM is illustrated in Figure 5 demonstrating the main modules developed in the implementation. We generally assume that input data to the framework is a set of workloads each having entries of sensitive attributes of the two entities passed to the matching model, the outcome of the model and the ground-truth. For easier use, we have integrated a built-in converter in FairEM to convert the benchmark data sets *Magellan* and *WDC* to the accepted format by Ditto and DeepMatcher. While DeepMatcher, outputs such as prediction results are stored in `pandas.DataFrames`, Ditto generates JSON outputs that need to be manually parsed. FairEM presents built-in parse functionality for Ditto outputs so that no further preprocessing by the user is required. Initially, each workload is examined to extract the set of all possible groups for the sensitive attributes. Next, for each entity, FairEM creates a group encoding based on the discussion of § 3:

- *Single attribute with binary values:* A vector of size two is created, with each element representing one of the demographic groups. The corresponding value to the demographic group of the entity is set to be 1 and the other value would be 0, e.g. for sensitive attribute *gender={male, female}*, an entity belonging to *male* demographic group is encoded as $\langle 1, 0 \rangle$ while a *female* entity is encoded as $\langle 0, 1 \rangle$.
- *Single attribute with multiple (exclusive) values:* Consider n as the cardinality of sensitive attribute of interest. A vector of size n is created and the corresponding value to the demographic group of the entity is set to be 1 while all the other elements will be 0, e.g. for sensitive attribute `race={Black, White, Hispanic, Asian, other}`, an entity belonging to `Black` demographic group is encoded as $\langle 1, 0, 0, 0, 0 \rangle$.
- *Single setwise attribute:* Similar to the previous case, a vector of size n is created, however this time, multiple elements in the vector can be equal to 1 since different demographic groups can happen at the same time, e.g. for sensitive attribute `genre={Rock, Pop, Jazz, Rap}`, an entity belonging to `Rock, Rap` demographic group is encoded as $\langle 1, 0, 0, 1 \rangle$.
- *Multiple attribute:*

Having encoded the sensitive attribute values for all entities in the dataset, next, FairEM moves to evaluation of the model with respect to the subgroups of interest. Users may not have a specific set of subgroups that they need to be concerned with, therefore, FairEM offers the capability to create all possible subgroups of

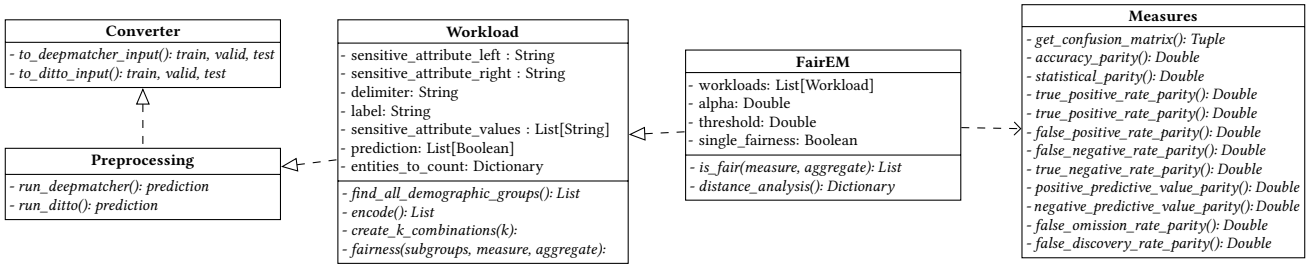

Figure 5: Class diagram for FairEM.

level-k in the subgroup hierarchy and investigate the performance of the matcher on them.

With the subgroups of interest discovered and enumerated, FairEM then evaluates each with regards to fairness measures of interest (discussed in section § 4.2) by calling *is_fair()* function. The *is_fair()* function calculates the disparities between the subgroup and its counterparts and if the values exceeds the user's fairness threshold constraints, subgroup will be reported as unfair. Depending on the number of workloads and aggregation functions, statistical analysis and hypothesis testing may have to be performed on the disparity results. Finally, FairEM visualizes the results in tables and heat maps, demonstrating a side-by-side overview of matcher's performance for each subgroup with respect to different fairness measures.

In addition, FairEM provides distance-based explanations when users are interested in investigating the reasoning behind unfair behavior of a matcher on a specific subgroup on a specific measure. Since in some cases the number of distances could potentially become large while each distance represents few subgroups and tuples, we execute equi-width binning on the subgroups such that k bins of approximately equal sizes are generated. Finally, for each unfair subgroup, average disparity values for each distance bin is computed and visualized in line charts, demonstrating matcher's behavior as similarities diminish.

7 EXPERIMENTS

We conduct comprehensive experiments on real-world data sets to validate FairEM and evaluate the fairness of two state-of-the-art entity matchers.

7.1 Experimental Setup

The experiments were conducted using a 3.5 GHz Intel Core i9 processor, 128 GB memory, running Ubuntu. The framework was implemented in Python.

7.1.1 Data Sets. For evaluation purposes, we used three real-world benchmark data sets:

- **Magellan iTunes-Amazon Data Set:** This dataset contains "structured" music data from iTunes and Amazon with a size of 539 instances, 132 of which being a true match [34]. We consider *genre* as a single setwise attribute sensitive attribute.
- **Magellan DBLP-ACM Data Set:** This dataset contains "structured" publication data from DBLP and ACM with a size of 12,363 instances, 2,220 of which being a true match [34]. We consider *venue* as the sensitive attribute with multiple values.
- **WDC Shoes Data Set:** Sub-sampled from WDC product data corpus and gold standard for large-scale product matching, this "textual" data set contains 42,429 instances of e-commerce product offering pairs from shoe domain, 4,141 of which being a true match. Each entity is associate with a *locale* at the end of the title that we extract and add as a separate column to use it as the sensitive attribute [38].

7.1.2 Entity Matching Frameworks. We employ Ditto and DeepMatcher frameworks to execute the entity matching task and we evaluate their performance on the aforementioned data sets.

7.2 Evaluation Plan

To evaluate the functionality of FairEM, we perform two sets of experiments:

- **Measure vs. Subgroup:** We investigate the performance of matchers in terms of single and pairwise fairness over a single or multiple workloads for all valid subgroups in the data sets.
- **Measure vs. Distance:** Once unfair subgroups are identified, we investigate the behavior of matchers as similarities between subgroups diminish.

Next, we choose a number of the identified discriminated subgroups from different settings and perform a case study on them to investigate the reasoning of FairEM for the unfair behavior of matchers.

7.2.1 Training Matchers. For DeepMatcher, we trained the model in 15 epochs and validated the outcomes with similarity threshold of 0.7. We choose *fastText* [9] pre-trained character-level embedding trained on English corpora for DeepMatcher due to its superior performance as reported in [34].

For Ditto, we trained the model with batch size and max sequence length of 64 and the learning rate of 3×10^{-5} in 40 epochs using *DistilBERT base (uncased)* [45] language model only pre-trained on English corpora. For optimization purposes, data was augmented using the deletion operator which randomly erases a span of tokens except for special tokens e.g. [COL] or [VAL]. Next, sequences were summarized by retaining only the high TF-IDF tokens. The resulting sequence will be of length no more than the max sequence length Finally, *general* domain knowledge was injected to the input sequences by tagging informative spans by inserting special tokens (e.g. PERSON) and normalizing certain spans such as numbers.

As for the fairness threshold, we follow EEOC's 80% rule [12], that is only 20% disparity is tolerated. We observe that distances

between entites for unfair groups in iTunes-Amazon were in the range of [1,11], which we binned into 4 equi-width bins.

7.3 Case Studies

Here, we highlight interesting observations from auditing DeepMatcher and Ditto. Figure 15 and 16 present a detailed view on the disparity values of various measures for DeepMatcher and Ditto on the iTunes – Amazon data set. More comprehensive results can be found in our technical report (submitted as supplementary material).

7.3.1 Case Study 1: iTunes – Amazon's Accuracy Parity. In our first case study, we evaluated Ditto and DeepMatcher for fairness on Magellan iTunes-Amazon dataset. In particular, we are interested to see if the two matchers have accuracy parity unfairness on any of the groups. As shown in Figures 6 and 7, both matchers were unfair towards the `R&B/Soul` genre – reflecting potential bias towards the artists and fans of this music genre. To further investigate the reasons behind the unfairness, we used the explanation techniques proposed in § 5.2. First, we considered other fairness measures to explain accuracy disparity. Considering Figure 6, it turns out DeepMatcher is fair for `R&B/Soul` on FPRP (and TNRP), while it is unfair on TPRP (and FNRP). This means the accuracy disparity for DeepMatcher is due to the disparity on TPRP. In other words, DeepMatcher mistakenly detected many of the pairs of `R&B/Soul` that are match as non-match, which resulted in a disparate accuracy for that group. On the other hand, the other fairness measures did not provide any explanation for the unfairness of Ditto for `R&B/Soul`.

An additional explanation can be sought by looking at the Distance vs. Accuracy plot in Figures 13 and 14. These figures compare the overall accuracy trend of matchers with respect to `R&B/Soul` accuracy trend as similarities between the groups diminish. On average, the matcher is performing well and although for `R&B/Soul`, both Ditto and DeepMatcher perform satisfactory for distance bin 2, for the more similar subgroups (bin 1) and less similar ones (bin 3), model significantly performs worse than the average explaining the existing unfairness with respect to AP.

7.3.2 Case Study 2: iTunes-Amazon – Ditto *v.s.* DeepMatcher. In this section, we continue with evaluating Ditto and DeepMatcher on different groups and different fairness measures, with the goal of comparing the matchers and find behavioral details causing unfairness. As observed in Figures 6 and 7, both models are unfair with respect to FPRP towards `R&B/Soul, Contemporary Country` and `Country` genres. By looking more closely at data, ground truth labels, and predicted labels, we observed that high FPRP is due to the high semantic and syntactic similarity of titles of song entities. A pair of songs mistakenly detected as a match by Ditto is the following:

Left song	*L-artist*	*Right song*	*R-artist*	y	h
`Tequila Loves Me`	`K. Chesney`	`Likes Me`	`K. Chesney`	'N'	'M'

First, both songs are by Kenny Chesney. But more importantly, using a pretrained language model, `Likes Me` and `Loves Me` are considered (almost) identical. As a result the model mistakenly labeled the left and right songs as match. Interestingly, such cases happen to be more frequent in genres such `Country`, resulting in FPRP unfairness for those groups.

One unfairness case in DeepMatcher (but not in Ditto) is TPRP (and FNRP) for groups such as `Dance, Electronic`, and `Rap`. Comparing DeepMatcher and Ditto results on the entries that DeepMatcher fails for, we interestingly identify match *pairs with identical titles that* DeepMatcher *mistakenly detected as non-match*. Ditto could successfully label those pairs as match because it assigns different importance to different features. As a result, considering song title as a strong match indicator, it could detect the match. DeepMatcher, on the other hand, merging the features together, considers equal importance for different features, hence, mislabeled the match pairs with identical titles due to small differences in other attributes. The high frequency of these cases in genres such as `Dance` causes the TPRP in DeepMatcher. Moreveor, Ditto serializes each entity as one input with structural tags intact and then calculates the similarity between the two, while DeepMatcher calculates the similarities between the two entities based on the corresponding features (title vs. title) and then aggregates all the similarities in the next step which can lead to a more uniform weight for all attributes rather than focusing on the important ones.

Finally, Ditto was unfair for *Pop* on FPRP, while DeepMatcher was fair. The reason for this was the popularity of different versions of songs with very similar titles which are considered as not match. The following is an example of such a pair:

Left song	*L-artist*	*L-album*	*Right song*	*R-artist*	*R-album*	y	h
`The Blood (Karaoke)`	`Taylor Swift`	`Karaoke`	`The Blood (Original)`	`Taylor Swift`	`Bastille`	'N'	'M'

Putting a high weight on the song title, Ditto mistakenly detected the two songs as match, while DeepMatcher, putting equal weights on different attributes, detected them as unmatch, considering the album title differences.

7.3.3 Case Study 3: Shoes Data set Non-English Subgroups. We evaluated Ditto and DeepMatcher on WDC Shoes dataset, both on single (Figures 8 and 9) and pairwise (Figures 10 and 11) fairness notions. As reflected in Figures 8 and 9, in general, both models are unfair towards entries that do not belong to non-English languages *locale*, including `fr, it, de, pl, es`, and other minority languages (grouped as `others`). The unfairness of both models on these groups can be due to the lack of proper representation of these languages on the training data, known as lack of data coverage [4, 5]. However, investigating the training data, we realized the training data has proper coverage of these languages. On the other hand, these models use *pre-trained language models and word/character embeddings being mainly trained on general English corpora*, and hence fail when it comes to entries in other languages. In other words, unfairness in these language models have been transformed into the unfairness of the matchers.

7.3.4 Case Study 4: DBLP-ACM Multiple Workloads. We perform multiple-workload analysis on the DBLP-ACM data set. This data set is large enough to be able to generate reasonably sized workloads. We generated 40 workloads, each of size 30%, of the original test data set (734 entity pairs). Our objective is to see if the matchers have any unfairness for the (DB) venues present in the dataset: `VDLB, SIGMOD, VLDBJ, SIGMOD Records`, and `ACM TODS`. The results

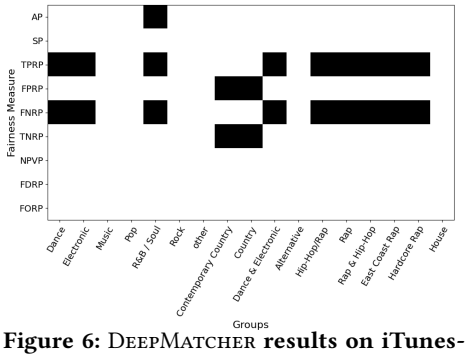

Figure 6: DeepMatcher **results on iTunes-Amazon data set: single fairness, 1 workload (*white* is fair, *black* is unfair).**

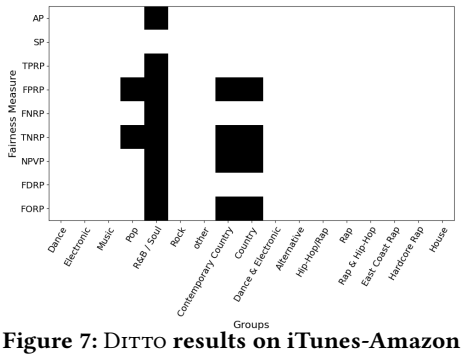

Figure 7: Ditto **results on iTunes-Amazon data set: single fairness, 1 workload.**

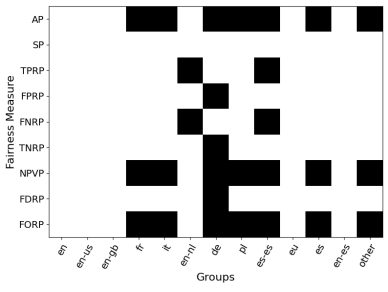

Figure 8: DeepMatcher **results on WDC shoes data set: single fairness, 1 workload.**

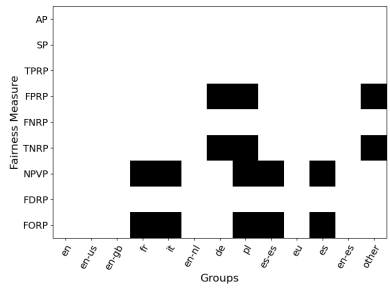

Figure 9: Ditto **results on WDC shoes data set: single fairness, 1 workload.**

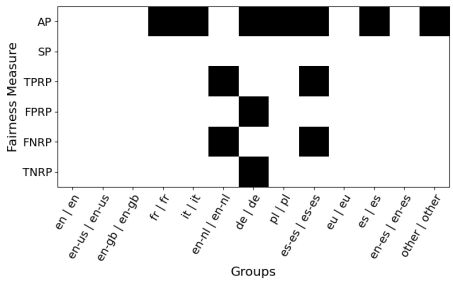

Figure 10: DeepMatcher **results on WDC shoes data set: pairwise fairness, 1 workload.**

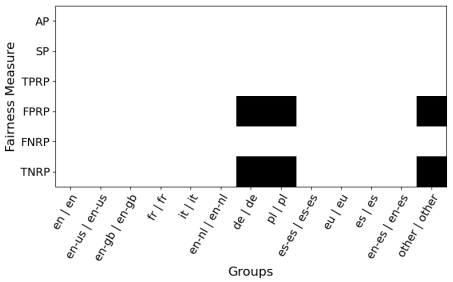

Figure 11: Ditto **results on WDC shoes data set: pairwise fairness, 1 workload.**

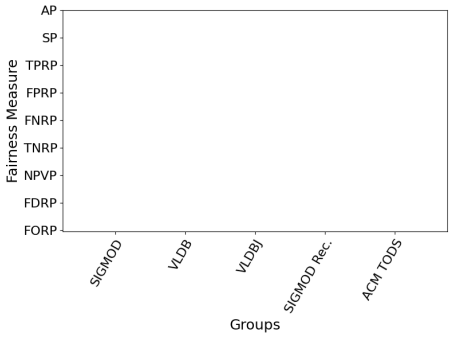

Figure 12: DeepMatcher **(also** Ditto**) results on DBLP-ACM data set: single fairness, 40 workloads.**

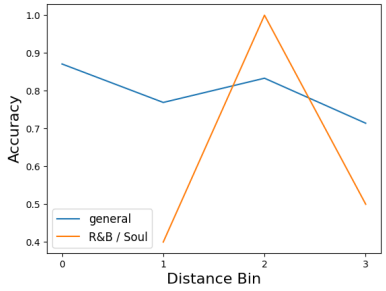

Figure 13: DeepMatcher **distance-based explanation for *R&B/Soul* subgroup in iTunes-Amazon data set.**

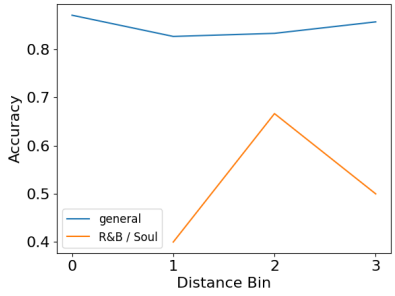

Figure 14: Ditto **distance-based explanation for *R&B/Soul* subgroup in iTunes-Amazon data set.**

are provided in Figure 12. Unlike previous cases, our experiments confirm that both matchers were fair for all venues, *on all fairness measures*. This confirms that the underlying data used for training the matchers was not biased with regard to the different venues. The training process also did not add "create" unfairness. As a result, using unbiased data results in the creation of unbiased matchers that are fair for all groups from all fairness perspectives.

8 RELATED WORK

Existing works on entity matching generally fall into one of the following three categories: 1) declarative rule 2) machine learning 3) crowd-sourcing based approaches. Rule-based methods such as [17, 48] specify rules for matching entities and have the advantage of being easily interpretable, however, a huge presence of domain experts is required in such contexts. Machine learning-based methods, mostly following the idea of [19], have adopted approaches such as SVM [11, 27], active learning [22, 24, 32] and clustering

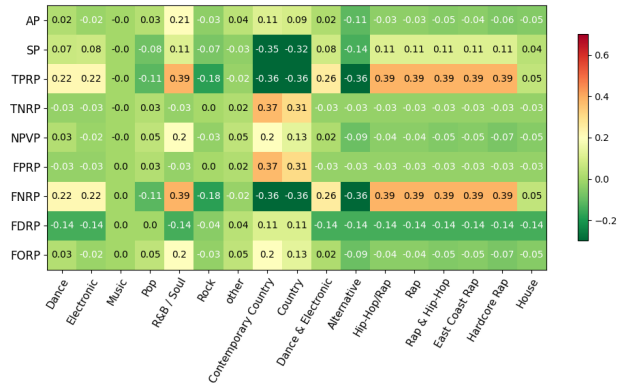

Figure 15: Disparity values for iTunes-Amazon subgroups with respect to different measures by DeepMatcher (values larger than 0.2 are considered to be unfair)

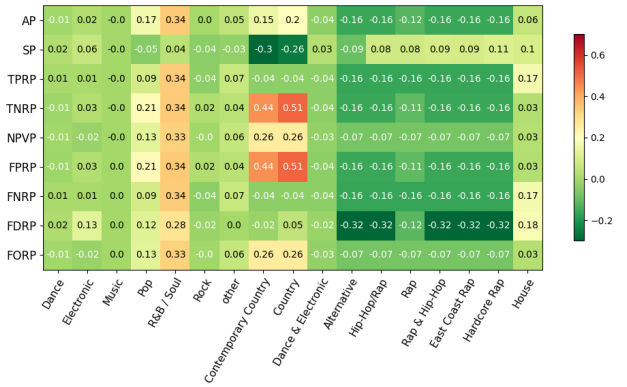

Figure 16: Disparity values for iTunes-Amazon subgroups with respect to different measures by Ditto

[39] to learn matching functions. Furthermore, deep learning techniques have recently shown promising results for entity matching tasks. DeepER [15] uses uni- and bi-directional RNNs with LSTM hidden units to convert each tuple to a distributed representation which can in turn be utilized to capture similarities between tuples. DeepMatcher [34] provides a categorization of deep learning solutions for entity matching (as SIF, RNN, attention and hybrid) and a categorization of entity matching problems (as structured, textual and dirty) in which deep learning can be effective. Ditto [29] treats the entity matching as a sequence-pair classification problem and uses pre-trained transformer-based language models and also further improves its performance by adopting optimization techniques such and domain knowledge injection, text summarization and data augmentation techniques. Finally, a line of work focuses on crowd-sourcing and using human experts knowledge for entity matching tasks [13, 21].

Auditing machine learning models for fairness has drawn a lot of attention in the recent years among different communities and many systems have been proposed for such applications. [41] is a bias and fairness audit toolkit that enables users to test models for several bias and fairness metrics in relation to multiple population sub-groups. [25] focuses on providing a solution for auditing models for intersectional fairness and mitigating bias. [54] proposes an explorative system for unfairness discovery, explanation and mitigation with the benefit of having human-in-the-loop. Additionally, since fairness does not have a unique definition, depending on the problem context, a great number of measures have been proposed for quantifying fairness [51].

Fairness has recently been studied in different steps of data pipeline, including data preparation [43, 44], data cleaning [46, 55], data integration [35, 36, 52], data profiling [20, 49], data bias identification [4, 5, 31], data acquisition [50], feature engineering [40] and query formulation [1, 2, 42, 47]. Fairness in entity resolution has briefly been studied in the literature. [16] proposes a constraint-based formulation approach to mitigate bias in entity resolution tasks as failing to address bias in these tasks may lead to systematic bias that jeopardize both accuracy and fairness of downstream data analysis. Fairness-aware entity resolution addresses discrepancies in the size of the groups in input data, where the majority of resolved entities belongs to a specific (advantaged) group.

To the best of our knowledge, we are the first to develop a system for auditing entity matching models for fairness and propose proper measures and comparison angles fitting the problem settings given the inherent differences with typical machine learning tasks that has so far been studied.

9 CONCLUSION

We proposed FairEM, a three-layer framework for auditing entity matching models for fairness. In data layer, we select and represent groups using a standardized encoding for unifying different types of sensitive attributes. Next, in the logic layer, we evaluate a workload of data with respect to applicable group fairness definitions specific to entity matching. Finally, in representation layer, we perform statistical evaluations on the fairness results and provide explanations for matcher's unfair behavior towards particular subgroups. We thoroughly assess two state-of-the-art matchers using FairEM with real world data sets and present and analyze the interesting observations. For future work, we intend to improve FairEM by adding a GUI, to facilitate easier investigation of the matchers. Since FairEM is not limited to the choice of matchers or data sets, we intend to integrate native support for more entity matchers and data sets specifically dirty data sets with missing data which can be quite challenging. Lastly, we would like to expand FairEM from solely auditing capabilities to a framework that offers unfairness resolutions.

REFERENCES

[1] Chiara Accinelli, Barbara Catania, Giovanna Guerrini, and Simone Minisi. 2021. The impact of rewriting on coverage constraint satisfaction.. In *EDBT/ICDT Workshops*.

[2] Chiara Accinelli, Simone Minisi, and Barbara Catania. 2020. Coverage-based Rewriting for Data Preparation.. In *EDBT/ICDT Workshops*.

[3] Abolfazl Asudeh and H. V. Jagadish. 2020. Fairly evaluating and scoring items in a data set. *PVLDB* 13, 12 (2020), 3445–3448.

[4] Abolfazl Asudeh, Zhongjun Jin, and HV Jagadish. 2019. Assessing and remedying coverage for a given dataset. In *ICDE*. IEEE, 554–565.

[5] Abolfazl Asudeh, Nima Shahbazi, Zhongjun Jin, and HV Jagadish. 2021. Identifying Insufficient Data Coverage for Ordinal Continuous-Valued Attributes. In *SIGMOD*. 129–141.

[6] Nils Barlaug and Jon Atle Gulla. 2021. Neural networks for entity matching: A survey. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 15, 3 (2021), 1–37.

[7] Solon Barocas, Moritz Hardt, and Arvind Narayanan. 2019. Fairness and machine learning: Limitations and opportunities. fairmlbook.org.

[8] Rachel KE Bellamy, Kuntal Dey, Michael Hind, Samuel C Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta,

Aleksandra Mojsilovic, et al. 2018. AI Fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias. *arXiv preprint arXiv:1810.01943* (2018).

[9] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606* (2016).

[10] Alexandra Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data* 5, 2 (2017), 153–163.

[11] Peter Christen. 2008. Automatic record linkage using seeded nearest neighbour and support vector machine classification. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 151–159.

[12] Equal Employment Opportunity Commission. 1979. The U.S. Uniform guidelines on employee selection procedures.

[13] Sanjib Das, Paul Suganthan GC, AnHai Doan, Jeffrey F Naughton, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, Vijay Raghavendra, and Youngchoon Park. 2017. Falcon: Scaling up hands-off crowdsourced entity matching to build cloud services. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 1431–1446.

[14] 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.

[15] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. 2017. DeepER–Deep Entity Resolution. *arXiv preprint arXiv:1710.00597* (2017).

[16] Vasilis Efthymiou, Kostas Stefanidis, Evaggelia Pitoura, and Vassilis Christophides. 2021. FairER: Entity Resolution With Fairness Constraints. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3004–3008.

[17] Wenfei Fan, Xibei Jia, Jianzhong Li, and Shuai Ma. 2009. Reasoning about record matching rules. *Proceedings of the VLDB Endowment* 2, 1 (2009), 407–418.

[18] 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.

[19] Ivan P Fellegi and Alan B Sunter. 1969. A theory for record linkage. *J. Amer. Statist. Assoc.* 64, 328 (1969), 1183–1210.

[20] 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.

[21] Chaitanya Gokhale, Sanjib Das, AnHai Doan, Jeffrey F Naughton, Narasimhan Rampalli, Jude Shavlik, and Xiaojin Zhu. 2014. Corleone: Hands-off crowdsourcing for entity matching. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. 601–612.

[22] Sairam Gurajada, Lucian Popa, Kun Qian, and Prithviraj Sen. 2019. Learning-based methods with human-in-the-loop for entity resolution. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2969–2970.

[23] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*. 3315–3323.

[24] Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. 2019. Low-resource deep entity resolution with transfer and active learning. *arXiv preprint arXiv:1906.08042* (2019).

[25] 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.

[26] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. 2016. Inherent trade-offs in the fair determination of risk scores. *arXiv preprint arXiv:1609.05807* (2016).

[27] Hanna Köpcke and Erhard Rahm. 2008. Training selection for tuning entity matching.. In *QDB/MUD*. 3–12.

[28] Hanna Köpcke and Erhard Rahm. 2010. Frameworks for entity matching: A comparison. *Data & Knowledge Engineering* 69, 2 (2010), 197–210.

[29] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. Deep entity matching with pre-trained language models. *arXiv preprint arXiv:2004.00584* (2020).

[30] Paul Pu Liang, Chiyu Wu, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2021. Towards Understanding and Mitigating Social Biases in Language Models. In *ICML (Proceedings of Machine Learning Research)*, Marina Meila and Tong Zhang (Eds.), Vol. 139. 6565–6576.

[31] Yin Lin, Yifan Guan, Abolfazl Asudeh, and HV Jagadish. 2020. Identifying insufficient data coverage in databases with multiple relations. *Proceedings of the VLDB Endowment* 13, 12 (2020), 2229–2242.

[32] Venkata Vamsikrishna Meduri, Lucian Popa, Prithviraj Sen, and Mohamed Sarwat. 2020. A comprehensive benchmark framework for active learning methods in entity matching. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 1133–1147.

[33] Christoph Molnar. 2020. *Interpretable machine learning*. Lulu. com.

[34] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 International Conference on Management of Data*. 19–34.

[35] Fatemeh Nargesian, Abolfazl Asudeh, and HV Jagadish. 2021. Tailoring data source distributions for fairness-aware data integration. *PVLDB* 14, 11 (2021), 2519–2532.

[36] Fatemeh Nargesian, Abolfazl Asudeh, and H. V. Jagadish. 2022. Responsible Data Integration: Next-generation Challenges. *SIGMOD* (2022).

[37] John Neter, Michael H Kutner, Christopher J Nachtsheim, William Wasserman, et al. 1996. Applied linear statistical models. (1996).

[38] Anna Primpeli, Ralph Peeters, and Christian Bizer. 2019. The WDC training dataset and gold standard for large-scale product matching. In *Companion Proceedings of The 2019 World Wide Web Conference*. 381–386.

[39] Alieh Saeedi, Eric Peukert, and Erhard Rahm. 2018. Using link features for entity clustering in knowledge graphs. In *European Semantic Web Conference*. Springer, 576–592.

[40] Ricardo Salazar, Felix Neutatz, and Ziawasch Abedjan. 2021. Automated feature engineering for algorithmic fairness. *PVLDB* 14, 9 (2021), 1694–1702.

[41] 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).

[42] Babak Salimi, Johannes Gehrke, and Dan Suciu. 2018. Bias in olap queries: Detection, explanation, and removal. In *Proceedings of the 2018 International Conference on Management of Data*. 1021–1035.

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

[44] Babak Salimi, Luke Rodriguez, Bill Howe, and Dan Suciu. 2019. Interventional Fairness: Causal Database Repair for Algorithmic Fairness. In *SIGMOD*. ACM, 793–810.

[45] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[46] Sebastian Schelter, Yuxuan He, Jatin Khilnani, and Julia Stoyanovich. 2020. FairPrep: Promoting Data to a First-Class Citizen in Studies on Fairness-Enhancing Interventions. In *EDBT*. 395–398.

[47] Suraj Shetiya, Ian P. Swift, Abolfazl Asudeh, and Gautam Das. 2022. Fairness-Aware Range Queries for Selecting Unbiased Data. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE.

[48] Rohit Singh, Vamsi Meduri, Ahmed Elmagarmid, Samuel Madden, Paolo Papotti, Jorge-Arnulfo Quiané-Ruiz, Armando Solar-Lezama, and Nan Tang. 2017. Generating concise entity matching rules. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 1635–1638.

[49] Chenkai Sun, Abolfazl Asudeh, HV Jagadish, Bill Howe, and Julia Stoyanovich. 2019. Mithralabel: Flexible dataset nutritional labels for responsible data science. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2893–2896.

[50] Ki Hyun Tae and Steven Euijong Whang. 2021. Slice tuner: A selective data acquisition framework for accurate and fair machine learning models. In *Proceedings of the 2021 International Conference on Management of Data*. 1771–1783.

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

[52] An Yan and Bill Howe. 2021. Equitensors: Learning fair integrations of heterogeneous urban data. In *Proceedings of the 2021 International Conference on Management of Data*. 2338–2347.

[53] Minghe Yu, Guoliang Li, Dong Deng, and Jianhua Feng. 2016. String similarity search and join: a survey. *Frontiers of Computer Science* 10, 3 (2016), 399–417.

[54] Hantian Zhang, Nima Shahbazi, Xu Chu, and Abolfazl Asudeh. 2021. FairRover: explorative model building for fair and responsible machine learning. In *Proceedings of the Fifth Workshop on Data Management for End-To-End Machine Learning*. 1–10.

[55] Yiliang Zhang and Qi Long. 2021. Assessing Fairness in the Presence of Missing Data. In *NeurIPS*.

APPENDIX

For the curious read, we include here the additional experiments and plots on DeepMatcher and Ditto.

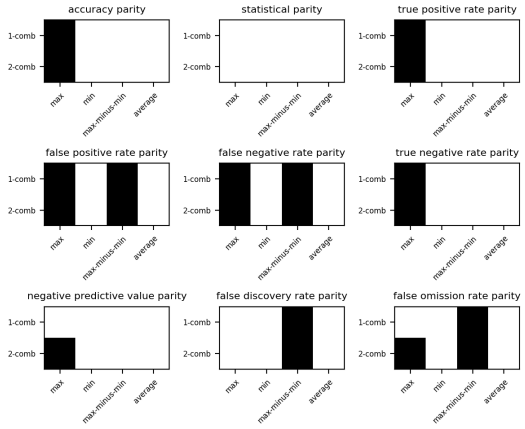

Figure 17: Aggregated unfairness results for 1-combination and 2-combination subgroups with respect to different measures by DeepMatcher

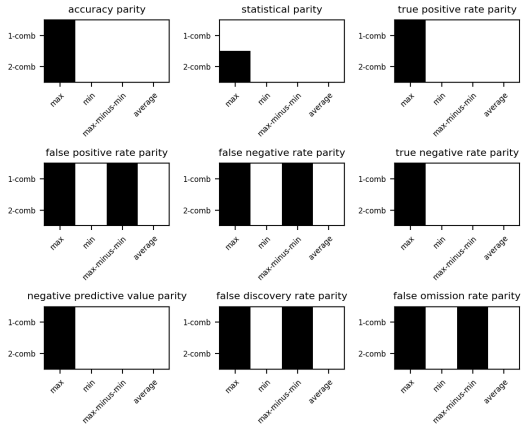

Figure 18: Aggregated unfairness results for 1-combination and 2-combination subgroups with respect to different measures by Ditto

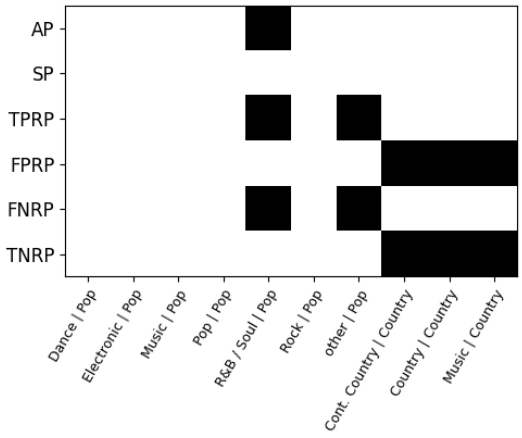

Figure 19: DeepMatcher results on iTunes-Amazon data set: pairwise fairness, 1 workload. (Only 10 subgroups shown due to space constraints)

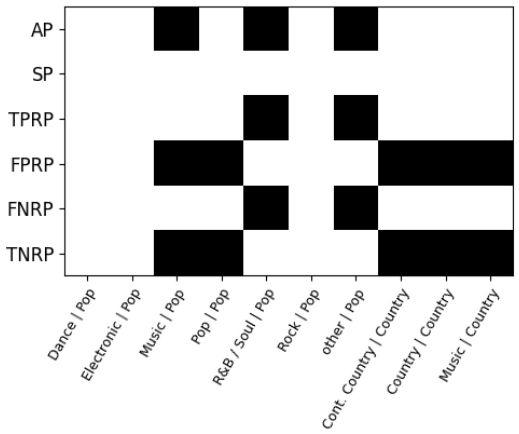

Figure 20: Ditto results on iTunes-Amazon data set: pairwise fairness, 1 workload.