8-23-2022 11:00 AM

Improving Deep Entity Resolution by Constraints

Soudeh Nilforoushan, *The University of Western Ontario*

Supervisor: Mostafa Milani, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science

Abstract

Entity resolution (ER) is the problem of finding duplicate data in a dataset and resolving possible differences and inconsistencies. ER is a long-standing data management and information retrieval problem and a core data integration and cleaning task.

There are diverse solutions for ER that apply rule-based techniques, pairwise binary classification, clustering, and probabilistic inference, among other techniques. Deep learning (DL) has been extensively used for ER and has shown competitive performance compared to conventional ER solutions. The state-of-the-art (SOTA) ER solutions using DL are based on pairwise comparison and binary classification. They transform pairs of records into a latent space that can be effectively compared to classify them as matched or unmatched. However, these techniques ignore possible constraints in record matching, including application-independent constraints (e.g., transitivity, symmetry, and reflexivity for matched records) and application-dependent constraints (e.g., cardinality constraints and fairness constraints).

In this thesis, I study constraints in SOTA deep ER solutions and integrate application-dependent and independent constraints with these solutions. I focus on transitivity, symmetry, and reflexivity as application-independent constraints and fairness constraints as application-dependent constraints. I present an algorithm that applies these constraints using data augmentation and shows this algorithm's effectiveness with real-world data.

Keywords: Entity Resolution, Deep Learning, Constraints, Fairness, Data Augmentation

Summary for Lay Audience

Entity resolution (ER) is the problem of finding duplicate data in a dataset and resolving possible inconsistencies and differences in this duplicate data. ER is one of the core tasks in data integration, where data from overlapping and possibly conflicting sources is integrated [4, 28]. It is also central to data quality assessment and cleaning, where erroneous duplicate records decrease data quality and hinder data usage. ER has been studied in several areas, including data management, information retrieval, machine learning (ML), artificial intelligence, and natural language processing (NLP).

Applications often collect data from heterogeneous sources where records have different features. This data heterogeneity can make finding relevant features for record comparison and resolution a daunting task.

Due to ER's technical challenges, many techniques have been developed to address them. One of the recent techniques is learning-based solutions and it consists of supervised and unsupervised machine learning (ML) that are used for ER. In this thesis, I focused on supervised learning. Supervised learning considers ER as a binary classification problem with pairwise record comparison where pairs of records are classified as matched and unmatched.

ER usually comes with semantic constraints that must be satisfied by ER solutions. For example, consider a dataset of patient records collected from two health institutions. If we assume each institution has unique patient records, an ER solution must match one record with at most one other. In this thesis, I focused on two types of constraints, fairness constraints, and equivalence constraints.

Acknowledgements

First and foremost I am extremely grateful to my supervisor, Dr. Milani for his valuable advice, continuous support, and patience during my MS.c. study. Without his assistance and dedicated involvement in every step throughout the process, this thesis would have never been accomplished.

I would also like to show gratitude to my committee members, Dr. Haque, Dr. Wang, Dr. Grolinger, and Dr. Lutfiyyah.

Finally, I would like to express my gratitude to my parents. Without their tremendous understanding and encouragement over the past few years, it would be impossible for me to complete my study.

Contents

List of Figures

List of Tables

List of Abbreviations

AUC area under the curve. 5

BERT Bidirectional Encoder Representations from Transformers. 3, 14

CRM customer relationship management. 13

DA data augmentation. 8, 16, 24

DistilBERT Distelled-BERT. 3

DL deep learning. ii, 3

ER entity resolution. ii, 2, 22

FPR false positive rate. 20

LM language model. 3, 14

LSTM long short term memory. 3

ML machine learning. 2

MPM multi perspective matching. 3

NLP natural language processing. 2

RNN recurrent neural network. 3

RoBERTa Robustly Optimized BERT Pre-training Approach. 3

ROC receiver operating characteristic. 6

SOTA state-of-the-art. ii, 22

SVM support vector machine. 3

TPR true positive rate. 20

Chapter 1

Introduction

Entity resolution (ER) is the problem of finding duplicate data in a dataset and resolving possible inconsistencies and differences in this duplicate data [43]. ER is one of the core tasks in data integration, where data from overlapping and possibly conflicting sources is integrated [4, 28]. It is also central to data quality assessment and cleaning, where erroneous duplicate records decrease data quality and hinder data usage. ER has been studied in several areas, including data management, information retrieval, machine learning (ML), artificial intelligence, natural language processing (NLP), and ironically the same problem, with minor differences, has been referred to by different names, such as entity matching, deduplication, record linkage, name resolution, object identification, and reference reconciliation [43]. While ER was initially considered for structured data (e.g., tabular data), its scope has moved toward semi-structured data (e.g., RDF, HTML, XML, and JSON) and unstructured data (e.g., text, sound, image, and video) [88, 13, 68].

ER is an essential stage of any data preparation pipeline and significantly impacts data-intensive applications that consume the high-quality data obtained from these pipelines for decision-making. The followings are a few examples, out of many, where ER can be consequential. ER is essential for studying population census and national surveys where duplicate records caused by possible changes in address, job, name, and marital status create data inconsistency and reduce the reliability and effectiveness of the data [2, 3]. Another example is online reviews (e.g., Amazon's reviews for products or Google's customer reviews for businesses), where duplicate business profiles can result in conflicting and unreliable reviews that can negatively impact businesses [32, 20]. Public health is another example where improving patient record matching is essential to ensure quality care [1]. Due to the importance of ER in these applications, practical and accurate ER solutions are always in high demand.

ER is considered a demanding task [43]. Applications often collect data from heterogeneous sources where records have different features. This data heterogeneity can make finding relevant features for record comparison and resolution a daunting task. Another ER challenge in many applications is erroneous data and low data quality. Erroneous data can make record comparison and matching more challenging as effective record comparison is contingent on having clean and error-free values. Perhaps the most important challenge in ER is its computational cost. Many ER solutions rely on feature-wise comparison between pairs of records, which can be costly and time-consuming when applied to large datasets.

Due to ER's technical challenges, many techniques have been developed to address it [89].

These techniques can be classified into a few categories. The earliest and perhaps the most straightforward techniques are distance-based or similarity-based and compare records based on some distance measure that combines the similarity, usually string similarity, between the records' attribute values and declare the records duplicates if they are close [80, 21]. Rule-based techniques (i.e. [34, 112, 105, 70, 86, 87]) use matching rules to find equivalent records, e.g., a matching rule states that two profile records belong to the same person if they have similar first and last names or the same SSN. The techniques in these two classes are simple, easy to implement, and do not require additional data. Still, they are limited in their effectiveness and applicability as they require matching rules or an effective comparison measure.

Both supervised and unsupervised ML are used for ER [4, 38, 22, 62, 10, 18, 109]. The unsupervised techniques usually apply clustering (e.g. hierarchical clustering [4, 38, 22] and correlation clustering [8, 24]) to form sets of equivalent records. The techniques based on supervised learning consider ER as a binary classification problem with pairwise record comparison where pairs of records are classified as matched and unmatched. Decision trees [62], support vector machines (SVMs) [10, 18], Bayesian networks [109] and other models are used for solving ER as a supervised learning problem. Probabilistic techniques [84, 36], crowdsourcing [26, 37, 44, 46, 110, 108], active learning [90, 58, 83, 96] are also integrated with ER solutions to improve their performance.

1.1 Deep Learning and Entity Resolution

Recently deep learning (DL) and deep neural networks have been extensively used for designing ER techniques that provide competitive performance (see [72] for a survey). These techniques also consider ER as a binary classification problem and use labeled training data to learn how to compare and match records. They can be divided into two categories based on whether they use pre-trained language models (LMs):

1. DeepER [30, 31] and DeepMatcher [81] are two early techniques that use deep neural networks, but they do not rely on LMs for word embedding. DeepER adopts recurrent neural network (RNN) with long short term memory (LSTM) hidden units to convert each record to a distributed representation using word embedding models such as GloVe [91] and fastText [12], and use the representation for pairwise comparison. DeepMatcher is an attention-based model similar to DeepER that considers the attribute-level similarities [81]. Other DL models were proposed that improve comparison for numerical values (multi perspective matching (MPM) [41]) or for heterogeneous data (Seq2SeqMatcher [85]), or improve accuracy with a token-level, attribute-level, and entity-level similarity calculation in a hierarchical architecture [40].

2. More recent techniques tune pre-trained LMs for ER classification tasks. `DITTO` fine-tunes Bidirectional Encoder Representations from Transformers (BERT), Distelled-BERT (DistilBERT), and Robustly Optimized BERT Pre-training Approach (RoBERTa) transformer-based machine learning techniques for NLP pre-training developed by Google, and shows that these knowledge in this LM can improve classification accuracy [71]. `DITTO` works by converting pairs of records to text and applying text classification to decide the pair match or does not match (see Section 3.1 for more detail about `DITTO`). Brunner and Stockinger [14]

extensively study the use of LMs for ER. They analyze different attention-based transformer architectures (e.g., BERT, XLNet [115], RoBERTa [74] and DistilBERT [100]), and compare their effectiveness in record matching.

In the experiments in Chapter 6, I use `DITTO` to evaluate the algorithm presented in this thesis for integrating constraints with ER. This has two reasons. First, it is one of the SOTA deep models with competitive matching quality in different program settings, e.g., with homogeneous and heterogeneous data [71]. Second, many other deep ER models use similar architectures with LMs [14], which means my experimental results can be generalized to other similar solutions. I briefly review some deep ER solutions together with the traditional ones in Section 2.3.

1.2 Constrained Entity Resolution

ER usually comes with semantic constraints that must be satisfied by ER solutions. For example, consider a dataset of patient records collected from two health institutions. If we assume each institution has unique patient records, an ER solution must match one record with at most one other. A similar constraint is used in [42] to improve the matching quality while extracting information from the web, where websites are expected to have unique records. As another example, consider matching employee records from multiple departments in an institution. If there is a known salary cap for the employees, the sum of the salaries in the records that are matched by an ER solution cannot exceed this cap. Note that this constraint is only applicable if the salary values are error-free. Aggregate constraints like this example are formalized and studied in [16].

The constraints in ER can be categorized into hard constraints and soft constraints. A soft constraint is likely to be satisfied; e.g., if two names are textually similar (e.g., ML King J and Martin Luther King Jr.), they are likely to refer to the same entity [104]. In contrast, hard constraints must be satisfied during record matching, e.g., while matching paper articles, if two articles match, their list of authors must also match. ER constraints can also be classified into application-dependent, such as the constraints mentioned so far, and application-independent constraints, such as symmetry, reflexivity, and transitivity of the matched records [113, 66, 6, 7].

ER constraints can express general user requirements in terms of quality record matching. For example, in critical applications where matching errors might have costly consequences, one might want to minimize errors as false positives, i.e., the record pairs that are incorrectly matched. One example is patient record matching, where incorrectly matching records can be more consequential compared to having duplicate patient records [17]. Another example is fairness constraints, where one might expect fair and equal matching quality between all subsets of a population [33]. For example, a model with a high overall accuracy might be unfair because it has low accuracy in some subpopulations, e.g., it might frequently mismatch patients with cancer or patients from a specific demographic. In some applications, one might prefer a fair model with a more balanced accuracy between all subpopulations, even if it has a lower overall accuracy for the entire population.

In this thesis, I incorporate constraints with SOTA deep ER solutions. I focus on two types of constraints: fairness constraints and the constraints for equivalence relations (i.e., reflexivity,

symmetry, and transitivity). In what follows, I use an example to explain these constraints and some of the related concepts.

Example 1.1 (Equivalence Constraints) Table 1.1 shows a few patient records from a relation that is collected from multiple health institutions. As such, it includes equivalent records that refer to the same patients. For instance, r_2, r_3, and r_7 that are highlighted with the same color are equivalent and refer to the same patient. Similarly, r_1 and r_6 are equivalent. However, r_4 and r_5, which are not highlighted, refer to different patients and are not equivalent.

An ER solution receives pairs of records (or record pairs) and labels them as *matched* or *unmatched*. For example, (r_2, r_3) is a record pair that might be correctly labeled as matched or incorrectly labeled unmatched. Sometimes, I refer to matched and unmatched by labels 1 and 0, respectively. The quality of an ER solution can be measured in terms of its classification accuracy, i.e., whether matched records are equivalent.

An ER solution defines a binary relation between the matched records. This relation is expected to be an equivalence relation that partitions the set of records that appear in a dataset into equivalent classes of matched records. Therefore, the binary relation must satisfy reflexivity, symmetry, and transitivity as the three properties of an equivalence relation. For reflexivity, an ER solution must match every record with itself, e.g., (r_1, r_1) must be labeled 1. Also symmetry implies that if (r_1, r_6) match, (r_6, r_1) must also match. Finally, transitivity means if (r_2, r_3) and (r_3, r_7) are matching pairs, then (r_2, r_7) must also match. This is the first type of constraint I study in this thesis, and I refer to them as the *equivalence constraints* throughout the thesis. ■

Fairness is recognized as an essential requirement for ML models that are used for automatic decision-making in sensitive applications [9, 98, 102]. Possible biases in these models can cause discrimination against specific individuals or subpopulations, particularly minorities. Fairness constraints usually require equal model quality for subpopulations, where the subpopulations are usually characterized by sensitive features, such as gender or race [77, 92]. There are different interpretations of equal model quality. For example, *equal opportunity* requires the same true positive rate between subpopulations, and *equalized odds* requires the same true positive rate and false positive rate [77]. Fairness constraints in [53, 39] use the area under the curve (AUC) to evaluate the quality of models with probabilities and measure the risk of classification (see Section 2.1 for more detail about fairness in ML).

This thesis presents a new fairness constraint for ER that limits biases in matching records across subpopulations. A meaningful measure of bias for record matching as a binary classification task is needed to formalize this constraint. I use the AUC that measures classification quality in terms of the risk of record matching when the ML models provide probabilities for matched and unmatched record pairs. This allows a deeper analysis of bias compared to the other notions of bias. My definition of bias using AUC is based on the recent measure of bias for binary classification in [53] and extends it for ER. Example 1.2 explains the application of AUC for measuring the quality of an ER solution. I will explain my AUC-based fairness in Example 1.3.

Example 1.2 (AUC for ER) The existing solutions that solve ER as a pairwise record matching usually provide a classifier that consists of a score function and a classification threshold.

The score function gets as input a record pair and returns a score, i.e., a real number that is usually in $[0, 1]$ that reflects the matching probability, where a higher score means a higher probability that the two input records are matched. The threshold decides whether a pair matches; if the score is higher than the threshold, the result is 1 and 0 otherwise. The score function can also be seen as a ranking function for record pairs based on their matching probability, where the threshold splits the ranked pairs to matched and unmatched. Tables 1.2a and 1.2b demonstrate two such score functions s_1 and s_2 that are applied the same record pairs from Table 1.1. The two lists show the same record pairs ordered by these scores. "Equiv" refers to the true labels, i.e., whether the two records in a pair are equivalent.

Using s_1 or s_2 and a classification threshold, we can decide matched and unmatched record pairs, e.g., using s_1 and a threshold 0.5, the bottom two pairs, (r_3, r_4) and (r_2, r_6), are unmatched and the other record pairs are matched. Both s_1 and s_2 are imperfect, independent of a threshold, as they rank one equivalent record pair lower than a nonequivalent pair, e.g., s_1 ranks (r_2, r_3) lower than (r_4, r_5). Nonetheless, the threshold controls the risk of false positives (matching nonequivalent records) and false negatives (missing the chance to match equivalent records); increasing the threshold decreases false positives and decreases false negatives.

A popular quality measure for classifiers is the AUC, the area under the curve, where the curve refers to the receiver operating characteristic (ROC) that plots true positive rate vs. false positive rate for varying classification thresholds in $[0, 1]$. This is a preferred measure where a classifier provides probabilities for different labels and gives a more meaningful measure of classification quality [35].

	NAME	**ETH**	**ADDRESS**
r_1	Goldie R. Chisolm	Caucasian	1195 Holly St. Athens, GA
r_2	T. S. Fatimata	Afr. Am.	109 Ralph St
r_3	Thokozani Fatimata	African American	109 Ralph St Belleville, NJ
r_4	Naomi Rodrigez	Asian	3672 Glory Road, FL
r_5	Xavier Rodrigez	Latino	3672 Glory Road, FL
r_6	G. R. Chisolm	Caucasian	1195 Holly St., GA
r_7	T Fatimata	African American	109 Ralph St Belleville

Table 1.1: Patient records

Pair	Equiv	Score
(r_1, r_6)	1	0.98
(r_4, r_5)	0	0.80
(r_2, r_3)	1	0.77
(r_3, r_4)	0	0.04
(r_2, r_6)	0	0.01

(a) s_1

Pair	Equiv	Score
(r_2, r_3)	1	0.91
(r_3, r_4)	0	0.86
(r_1, r_6)	1	0.79
(r_4, r_5)	0	0.12
(r_2, r_6)	0	0.07

(b) s_2

Table 1.2: Two score functions applied on the same list of record pairs.

For a score function (e.g., s_1 and s_2) in an ER problem, the AUC is the probability that an equivalent record pair is correctly ranked higher than a nonequivalent record pair. This is a

measure of how well the score function ranks record pairs based on their matching probability and can be used to evaluate the risk of entity matching using the score function, independent of any particular threshold. The AUC is 1 when the function ranks all equivalent pairs higher than nonequivalent pairs, it is 0.5 when ranking is random, and it is 0 when the score function ranks nonequivalent pairs higher than equivalent pairs.

To estimate the AUC of s_1 using the limited number of record pairs in Table 1.2a, one can compare the equivalent pairs (e.g., (r_1, r_6) and (r_2, r_3)) with the nonequivalent pairs (e.g., (r_4, r_5), (r_3, r_4), and (r_2, r_6)). In all these 6 comparisons, the record pairs are correctly ranked by s_1, except in one where the equivalent pair (r_2, r_3) with score 0.77 is ranked lower than the nonequivalent pair (r_4, r_5) with score 0.80. Therefore, the estimation of s_1's AUC is $5/6$. Following similar steps, one can obtain the same estimate for s_2 using the record pairs in Table 1.2a, which shows both score functions have the same classification quality based on their AUCs. ■

To measure bias by comparing accuracy across subpopulations, I define subAUC by extending the definition of AUC to subpopulations similar to [53]. In a nutshell, subAUC for a subpopulation is the probability that an equivalent record pair is ranked higher than a nonequivalent record pair when at least one record in the record pairs is from the subpopulation. The subAUC of a score function for a subpopulation measures how well the ranking specified by the function works for the records in the subpopulation. I explain the concept of AUC for a subpopulation and the notion of bias and fairness constraints in Example 1.3.

Example 1.3 (AUC-Based Fairness Constraints) The ethnicity attribute in Table 1.1 defines 4 subpopulations of Caucasians, African Americans, Asians, and Latinos. To estimate the subAUC of s_1 for Asians using Table 1.2, I consider (r_4, r_5) and (r_3, r_4) that are record pairs including r_4, the only record belonging to an Asian individual. Since the pairs are nonequivalent, they must be compared with equivalent pairs (r_1, r_6) and (r_2, r_3), making 4 possible comparisons. Out of these comparisons, s_1 only miss-ranks one of them as it ranks (r_4, r_5) higher than (r_2, r_3). Therefore, the subAUC for Asians is $3/4$. Similarly, the subAUC estimations for African Americans, Latinos, and Caucasians are $4/5$, $1/2$, and 1. This means the quality of s_1 differs between the subpopulations implicating biases in ER. This bias can be measured by the gap between the minimum AUC and the maximum AUC: $1 - 1/2 = 1/2$. For s_2, the estimated subAUCs for Asians, African American, Caucasians and Latinos w.r.t. Table 1.2b are respectively $3/4$, $4/5$, $4/5$, and 1, which means the bias is $1 - 3/4 = 1/4$. This shows although s_1 and s_2 have the same overall AUC (i.e. $5/6$), s_2 incurs less bias compared to s_1. I use this notion of bias to define the *fairness constraint* that requires an ER solution to have a bias lower than a specific value. For example, a fairness constraint that expects a bias lower than $1/2$ will reject s_1 but accept s_2 as a viable solution. ■

In Chapter 6, I run experiments using real-world data that show biases in `DITTO` as one of the SOTA deep ER solutions. I used the bias explained in Example 1.3 to evaluate the fairness of `DITTO` and found biases in ER for subpopulations. According to my analysis of the experimental results, there are two reasons for these biases. The first cause of bias is the difference in the data distribution in subpopulations. For example, the difference in the demographic or personal patient data, illustrated in our running example by the similarity in address and name

attributes, can lead to an ER solution performing differently for two subpopulations. Secondly, the size of training data for subpopulations can be different, resulting in poor performance due to over-fitting or under-fitting and causing biases (see Chapter 6).

1.3 Data Pre-processing for Incorporating Constraints

To apply the fairness and equivalence constraints, I introduce an algorithm that augments training data with new record pairs. The algorithm finds missing record pairs to satisfy the constraints to apply the equivalence constraints and adds them to training data. For the fairness constraints, I use `Rotom` [78], a general data augmentation (DA) framework for data management tasks, including data cleaning, to generate new equivalent record pairs. `Rotom` leverages Seq2Seq-based NLP models, and policies for selecting and weighting augmented data examples and balances between diversity and quality in DA (see Chapter 3 for a brief review of `DITTO` and `Rotom`). The experimental results in Chapter 6 show a decrease in biases in `DITTO` when applied for ER in three datasets. The results also demonstrate a decrease in bias in `DITTO` when applying the equivalence constraints. In this work, I consider both equivalence and fairness constraints as soft constraints and try to minimize bias and decrease the violations of the equivalence constraints.

This thesis is aligned with the recent research trend about Fairness in Data Cleaning and Data Preparation (e.g., [33, 57, 79, 11, 76, 101, 114]). Unlike most of the existing research on fairness in ML that focuses on the last stages of developing ML models (i.e., model training and tuning), these research studies consider the early stages of model development (i.e., data collection, preparation, and cleaning). I also note that this thesis concentrates on the first part of ER, namely finding duplicate data. The second part, about resolving possible inconsistencies in the duplicate data, is beyond the scope of this work.

1.4 Contributions and Thesis Structure

In summary, I make the following contributions in this thesis:

1. I formalize fairness constraints using a novel measure of bias for ER based on the AUC.
2. I define the problem of ER with fairness constraints and equivalence constraints.
3. I design an algorithm for DA that applies equivalence constraints and fairness constraints by adding new labeled record pairs to training data.
4. I conduct experiments to analyze bias in ER and also show the effectiveness of the algorithm in reducing bias and improving overall accuracy.

The thesis is structured as follows. I begin by explaining some background about ER and DA for data cleaning in Chapter 3. Then, in Chapter 4, I define ER with constraints where I also formalize AUC-based fairness and our new notion of bias in ER. In Chapter 5, I present an algorithm for reducing bias and incorporating the equivalence constraints with ER. I conduct experiments with real-world data in Chapter 6 where I show the effectiveness of the algorithm

in three different datasets. Chapter 2 is related work, and Chapter 7 consists of final discussions, conclusion, and future work.

Chapter 2

Related Work

This thesis is closely related to several research areas that I briefly review the closest ones in this chapter.

2.1 Fairness in Machine Learning

Fairness in ML is often characterized as having equal prediction quality, between individuals, i.e., individual fairness, or between subgroups or subpopulations, i.e., group fairness [29, 67]. This thesis concerns group fairness in the binary classification model. Some of the widely used notions of fairness for binary classification are the followings:

- *Equalized odds* requires equal true positive rate and false positive rate between subpopulations [50].
- *Equal opportunity* expects the subpopulations should have equal true positive rates [50].
- *Demographic parity* requires the likelihood of a positive outcome to be the same for all subpopulations [29, 67].
- *Fairness through awareness* says any similar individuals based on a task-specific similarity measure should be assigned to the same class [29].
- *Counterfactual fairness* expects each individual to be assigned to the same class in a counterfactual world where the individual belonged to a different demographic group [67].
- *AUC-based fairness* assesses the disparate impact of *risk* in binary classification and requires that the same probability of correctly ranking individuals from different subpopulations [39, 53]. In this thesis, I use this notion of bias to measure the risk of record matching. Other notions of bias can also be used for ER as a binary classification problem.

Biases causing unfair ML models can be classified into a few categories (see the survey [77] for more detail):

- *Measurement bias* happens when information collected for use as a study variable is inaccurate. For example, a faulty thermometer can cause biases if it fails only for certain groups of patients.

- *Representation or sampling bias* occurs when the process that samples data is biased, e.g., it only selects samples from certain groups.

- *Aggregation bias* arises when general false conclusions are made about a group that does not hold for individuals in the group.

Biases in ER can be in any of the categories above.

There are three categories of techniques to incorporate constraints with ML models, including deep neural networks that are used for solving ER:

- *Pre-processing:* The techniques in this category aim to modify the training data to minimize the bias existing in the model that is trained using the data [54, 103, 51]. The common techniques in this category include sampling or reweighing the training samples [54], repairing data and changing individual data records [48, 99], adversarial debiasing [116], and augmenting with synthetic data [103].

- *In-processing:* The techniques in this category incorporate fairness constraints with the learning procedure [56, 55]. This is done in different ways, e.g., by adding a regularizer to the loss function that balances the incurred loss in subpopulations [55], training a model per subpopulation [15], fair statistical inference [82], and causal reasoning [59]

- *Post-processing:* The techniques for post-processing see an ML model as a black box and mitigate bias by processing the output of the model [60, 75, 93, 94, 50]. These techniques are ideal for debiasing models in a runtime environment without the possibility of making changes to training data and training processes. The general idea of post-processing techniques is to modify the model outcome for a selected sample to guarantee overall fairness.

2.2 Constraints in Entity Resolution

There are several research studies that incorporate constraints with conventional ER solutions and are different from this work, which focuses on deep ER. Here, I review the main constrained ER solutions and the types of constraints considered in these solutions.

The authors in [16] address the problem of integrating aggregate constraints into ER. Aggregate constraints are hard semantic constraints on the aggregate values of the matched records in ER. The authors show that ER with these constraints is computationally expensive and requires searching in a large space of possible partitions. They formulated the problem as a maximum constraint satisfaction problem and leveraged textual similarity to restrict the search space of partitions. They present an algorithm that optimally solves the constraint maximization problem over this search space, and they conduct experiments over real data that show leveraging aggregate constraints can significantly improve the accuracy of ER solutions.

The ER solution in [104] exploits a broad range of application-dependent constraints to significantly improve matching accuracy. An example of an application-dependent soft constraint while matching research scholars says if two researcher profiles have similar names and share some co-authors, they are likely the same. The solution is based on a probabilistic interpretation of the constraints that allows incorporating soft and hard constraints.

Deduplog [5] is a declarative language based on *Datalog* for expressing application-dependent constraints in ER. Datalog is a declarative rule-based language for writing recursive queries and constraints on relational databases. An example of a constraint that can be expressed by Deduplog is "each paper has a unique venue," which means if two paper references are pointing to the same paper, then their conference references also refer to the same venue. Deduplog allows collective ER in a relational database, which means it applies deduplication for multiple relations, e.g., both papers and conferences. The authors use a new clustering algorithm on graphs, called clustering graphs, to evaluate Dedupalog programs.

The closest work to this thesis is FairER [33], which also considers fairness constraints with ER. The fairness constraints in FairER are different from those defined in this thesis and are limited to ER for matching records collected from two sources. FairER sorts pairs of records from the two sources based on their matching probability and additionally expects the same number of records from each subpopulation to appear in top k pairs.

Gemmel et al. [42] use global constraints to improve the quality of matching entities from websites. The constraints are based on the socio-economic property that websites such as Netflix and IMDB try to avoid duplicate entities within their websites, as duplicates can negatively affect their services. Considering this while matching records from one website with records from another website can improve the matching property.

2.3 Other Entity Resolution Techniques

In Chapter 1, I discussed some of the main ER solutions. In this section, I briefly review some other ER solutions and the techniques to improve the existing ER solutions.

Crowdsourcing is a technique to involve human cognitive abilities in solving problems that cannot be effectively solved with completely automatic techniques [69]. Crowdsourcing is usually implemented using online frameworks, e.g., Amazon Mechanical Turk[1] and CrowdFlower [2]. Crowdsourcing is extensively used for solving ER in frameworks, such as Corleone [45], Falcone [23], Crowder [111], and Zencrowd [25].

Active learning is a solution for reducing the number of pairs that need to be labeled by actively selecting the most informative examples [97, 95]. ER tasks have limited access to labeled data and would require substantial labeling effort upfront, before the actual learning of the ER models. The active learning algorithm searches for high-confidence examples and uncertain examples, which provide a guided way to improve the precision and recall of the transferred model to the target dataset. This paper [58] combines transfer learning and active learning to prevent decreasing the performance of DL models with few labels

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. This paper [52] introduce a solution

[1] https://www.mturk.com/

[2] https://visit.figure-eight.com/People-Powered-Data-Enrichment_T

that a transfer learning framework leverages both the labeled and massive unlabeled data to train the model. Auto-EM [117] is a self service and automatic system with little or no training data. Auto-EM is an important operator in general purpose platform such as commercial customer relationship management (CRM) system, where an aspiration is to allow enterprise customers using the CRM system to automatically match their customer records across data silos in enterprises [107, 61].

Chapter 3

Background

In this chapter, I first review in Section 3.1 DITTO [71] as a SOTA deep ER solution that I use in my experiments, and then, in Section 3.2, I briefly explain Rotom [78], the DA framework that I apply in my pre-processing algorithm.

3.1 Entity Resolution using Language Models

DITTO is a novel ER solution based on the language models (LMs). DITTO [71] uses transfer learning by fine-tuning pre-trained LMs, which are more powerful in language understanding than traditional word embeddings. It also improves the matching with three optimizations:

- It emphasizes some important parts of the input data set by adding domain knowledge.
- It summarizes long string to the most essential is retrained and used for ER.
- It augments training data with difficult examples, which learns the model to learn beyond the existing training data and also reduces the size of training data [71].

I provide the main concepts behind ER. An end-to-end ER systems consists of *blocker* and *matcher*. Record pairs comparison is expensive since it is a Cartesian product of database records. Blocking is used to solve this problem and it is the first step that discards as many comparisons as possible without missing any matches. In other words, blocking places similar entities in the same blocks so it avoids extra comparison for the entities that are unlikely to match. The entities that co-occur in at least one block are compared during matching. This applies a combination of string similarity measures to the values of selected attribute names. Without these two steps, the cost of entity matching would be $O(n^2)$, as every entity has to be compared with all others. By applying these two techniques the cost of pairwise matching reduces and the cost of pairwise matching depends on the structure of the dataset. Most ER approaches compare pairs of entities and determine binary match mappings consisting of all correspondences or links between two matching entities in each blocking. The resulting degree of similarity is then used to assign the entity pairs into one of the three possible categories, i.e., match, non-match, or uncertain [19].

DITTO is one of the first ER solutions which leverages LMs and can be used with three pre-trained LMs: BERT [27], DistilBERT [100] and RoBERTa [74]. BERT is a transformer

for pre-training over many unlabeled textual data to learn a language representation that can be used to fine-tune specific machine learning tasks. DistilBERT is a small, fast, cheap, and light transformer model trained by distilling(approximating) the BERT base. It has 40% fewer parameters than BERT and runs 60% faster while preserving over 95% of BERT's performances. RoBERTa, is retraining of BERT with improved training methodology, 1000% more data, and compute power.

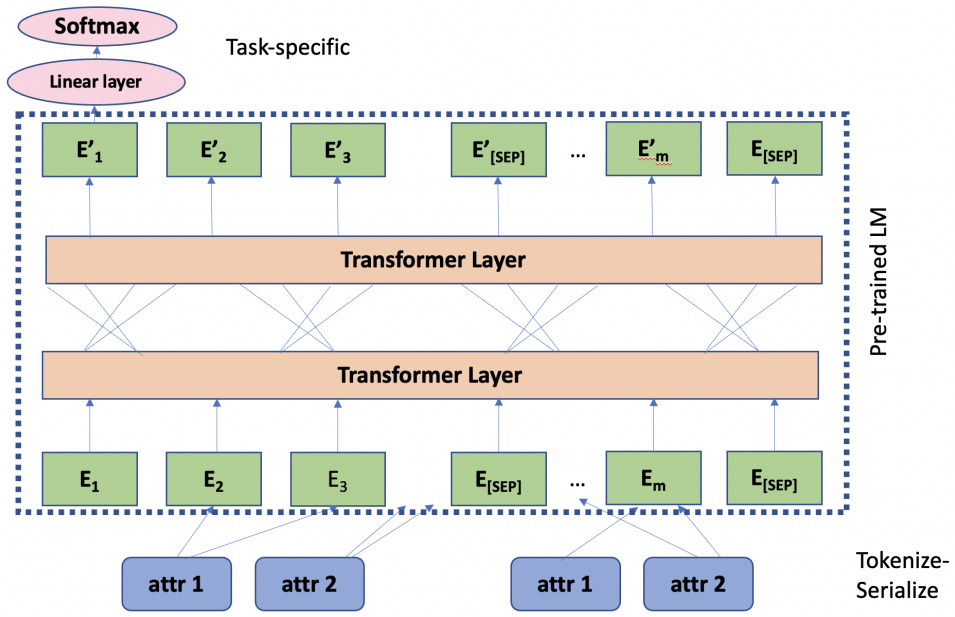


Figure 3.1: The architecture of DITTO ([71])

Figure 3.1shows the architecture of DITTO. It consists of three layers, Tokenize-Serialize, Pre-trained LM, and Task-Specific.

- *Tokenize-Serialize:* It is the input layer that serializes two entries and feeds them to the model. DITTO needs to convert the candidate pairs into a sequence of tokens to be interpreted by the model. DITTO serializes entries as follows

  ```
  record = [COL] attr_1 [VAL] val_1 ... [COL] attr_k [VAL] val_k
  ```

  where [COL] and [VAL] are special tokens that specify column (attribute) names and their values. For example, the followings are two serialized entries with the same attributes, name, address, and ethnicity:

  ```
  r1 = [COL] Name [VAL] Thomas Bartkow [COL] Address [VAL] Somerville
       street [COL] Ethnicity [VAL] Asian
  r2 = [COL] Name [VAL] Mackenzi Lansdell [COL] Address [VAL]
       Lazarusplace [COL] Ethnicity [VAL] African
  ```

  A record pair consists of two records and a label separated with a special token [SEP] as the following example shows:

```
r1 [SEP] r2 [SEP] 0
```

In this example, the two records are not matched and labeled with 0. The format of the dataset that I used is similar to this example.

- *Pre-trained LM:* The pre-trained LM consists of token embedding and transformer layers such as BERT. The transformer calculates token embeddings from all the tokens in the input sequence and then the embeddings which are generated can capture the semantic and contextual understandings of the words. Such embeddings can recognize that the same word may have different meanings in different phrases. For example, the word *sharp* has different meanings in *sharp resolution* versus *sharp TV*. Pre-trained LMs will embed *sharp* differently depending on the context while traditional word embedding techniques such as fastText always produce the same vector independent of the context.

- *Task specific layer:* DITTO add a fully connected layer and a softmax output layer for binary classification for adding labels. Softmax normalizes an input value into a vector of values that follows a probability distribution whose total sums up to 1. Then, it turns these values, into binary numbers, if the output of softmax is less than 0.5 it converts it to 0 otherwise, it converts it to 1, since we choose 0.5 as the threshold of softmax.

3.2 Data Augmentation for Entity Resolution

Rotom is a multi-purposed data augmentation (DA) framework for training high-quality machine learning models while requiring only a small number of labeled examples. Rotom has a simple task formulation of sequence classification so that it covers a wide range of data management and NLP tasks including ER, error detection in data cleaning, text classification, and more.

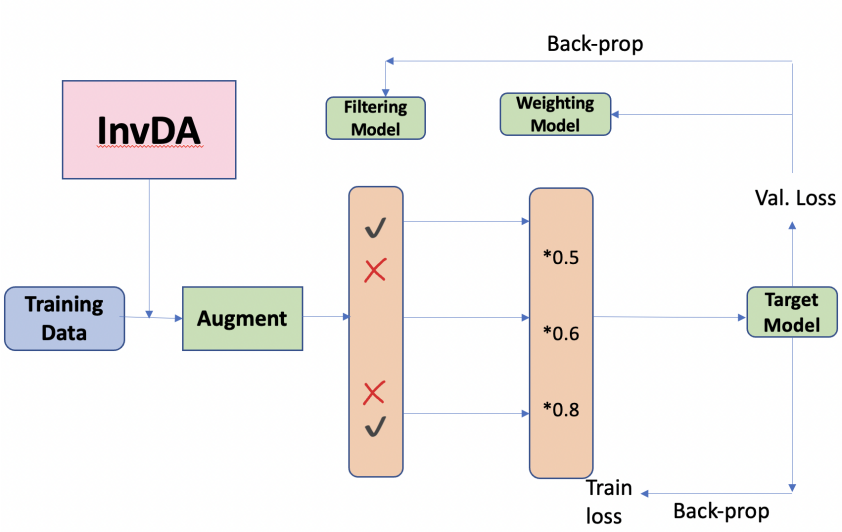


Figure 3.2: The architecture of Rotom [78]

DA is a technique to generate additional labeled examples from existing labels for training the model [73]. Since gathering labeled data for training DL models is costly, DA became an essential task in deep learning models. Although DA is effective, it risks changing the ground truth labels. So, the corrupted labels can damage the learning process. For example, consider this sentence, "Where is the orange bowl?" by changing only "where" to "what"; the intent of this sentence can be completely changed. If we limit DA to change the words to a similar one, this can also be useless for training the DL model, and it fails to learn much additional information, for example, "Where is orange bowl?", this sentence by removing "the" is not diverse, but "Where is the orangish bowl?" is a good replacement. `Rotom` tune the trade-off between the diversity of generating labels and quality for data augmentation and addresses this challenge by the following efforts:

- Introducing a new DA operator, InvDA, which can generate diverse yet natural examples by formulating DA as a Seq2Seq problem.
- Changing a new filtering model that chooses the good augmented examples among the whole augmented examples.

There is also another challenge, the models need to retrain and keep trying combinations until the result is satisfying, and this process is inefficient. For example, in word replacement, we need to decide which words to replace and how to sample the target words. If there is more than one operator, the developer needs to pick an operator each time and retrain the model then check the result and decide whether continue this process or not. This situation can be more difficult when we combine multiple operators, which significantly increases the search space. `Rotom` address this challenge by combining augmented examples generated by multiple DA operators.

Figure 3.2 shows the architecture of `Rotom` which consists of three layers:

- *Input:* `Rotom` supports DA for different tasks, including error detection and ER. The input is a sequence of tokens, while its content depends on the task. I use `Rotom` for ER, where the input represents a record pair.
- *InvDA:* The InvDA operator learns a Seq2Seq [106], DA model, by reconstructing corrupted unlabeled sequences. In the Seq2seq task, an original sequence from the training set generates an augmented training sequence. InvDA can generate natural and diverse augmented sequences by formulating the task as Seq2Seq, which is arbitrarily different from the original sequence. The model's flexibility of `Rotom` is in formulating data as a sequence. Unlike other transformations that limit the number of changes to the original sequence, the output of Seq2Seq can be arbitrarily different from the input. However, training a high-quality Seq2Seq model usually requires many labeled sequence pairs. To overcome this label requirement, `Rotom` applies the idea of self-supervision.

  Table 3.1 shows an example that invDA generates natural and diverse augmentations such as converting "relational databases" from the original sentence to "databases", "database systems" or "open-source databases". On the other hand, we observe that sequences generated by simple transformations (DA1 and DA2, two simple augmentation operators in `Rotom`) are somewhat unnatural or deviate from the original meaning.

- *Meta-Learning framework:* Although invDA or other operators apply augmentation, there is still no guarantee that they improve the performance of machine learning, Rotom develop a *meta-learning* and semi-supervised framework that selects and combines augmented data.

  This framework consists of three models, the *filtering model*, the *weighting model* and the *target model*. The filtering model chooses the best examples among the pool of augmented examples. Then, the weighting model assigns weights to the remaining examples, assembles them, and feeds them to the training target model.

  The idea of meta-learning is learning from the past and training the model; the filtering and the weighting models are gradually learning how to generate training batches of higher quality by descending to a low validation loss.

	ER-DBLP-ACM paper matching
Original	[COL] title [VAL] effective timestamping in relational databases
DA1	[COL] title [VAL] effective _ in relational databases
DA2	[COL] title [VAL] effective relational in databases timestamping
InvDA1	[COL] title [VAL] effective timestamping in databases
InvDA2	[COL] title [VAL] effective timestamping in database systems
InvDA3	[COL] title [VAL] effective timestamping in open-source databases

Table 3.1: Example of InvDA for ER. The _ symbol indicates a deleted token [78, Table 5]

Chapter 4

Problem Definition

In this chapter, I define the problem of ER with constraints (Section 4.1), and formalize equivalence constraints (Section 4.2) and fairness constraints (Section 4.3).

4.1 Entity Resolution with Constraints

I start with a training dataset $D = \{(r_1, t_1, l_1), ..., (r_n, t_n, l_n)\}$ that consists of record-pairs (r_i, t_i) and their labels l_i. If r_i and t_i are *equivalent*, i.e., they refer to the same real-world entity, then $l_i = 1$, and $l_i = 0$ otherwise. I assume r_i and t_i are from the same schema with features $A_1, ..., A_m$. I also assume all the records that appear in pairs in D are from a set of possible records R. For a records $r \in R$ and a feature A, $r[A]$ refers to the values of the feature A in the record r. Given a sensitive feature $A \in \{A_1, ..., A_m\}$ and a sensitive value a in the domain of A, $R_{A,a} = \{r \in R \mid r[A] = a\}$ is a *subgroup* or *subpopulation* in R. Intuitively, a subpopulation is a subset of records in R with the same sensitive values.

A binary classifier for ER is as a function $f : R \times R \mapsto \{0, 1\}$ that receives a record-pair and returns 0 or 1. If $f(r_i, t_i) = 1$, I say the classifier f matches the record-pair (r_i, t_i), or the records r_i and t_i match when the classifier is clear from context. I call l_i the true label of (r_i, t_i), and use $\hat{l}_i = f(r_i, t_i)$ to refer to the predicted label of the pair. The problem of *Entity Resolution (ER in short)* is to find the best classifier w.r.t. a given training dataset D. Different notions of loss for binary classification can be used to define the best classifier for the ER problem while comparing the predicted labels with the true labels in the training dataset D.

Given a test dataset in the same form as D and with the records in R, the quality of an ER solution f depends on whether the matched record pairs by f are equivalent. In other words, it depends on whether f is accurate as a classifier (i.e., $\hat{l}_i = l_i$). One can use precision, recall, and F1 measure or other accuracy measure based on the confusion matrix to evaluate the quality of f for ER.

4.2 Equivalence Constraints

As I formulated above, ER is usually solved as a pairwise record-matching problem. This is mainly because binary classifiers provide the best performance in ER. However, the ER problem is a data partitioning problem where the goal is to partition a set of records into classes

of equivalent records. By considering ER as a partitioning problem, unsupervised learning is used for clustering records where clustering measures are employed to evaluate the quality of an ER solution [4, 38, 22]. In this problem setting, any ER solution trivially satisfies symmetry, reflexivity, and transitivity as the properties of an equivalence matching relation.

The existing ER solutions for pairwise record matching often ignore these properties of the matching relation. Research studies show that incorporating these properties as matching constraints can increase the quality of record matching [88], non of which consider incorporating these constraints with deep ER solutions.

In my problem formulation, I incorporate the following equivalence constraints with the classifier f:

- *Reflexivity:* For any records r and t in R, $f(r, r) = 1$ and $f(t, t) = 1$.
- *Symmetry:* For any records r and t in R, if $f(r, t) = 1$ then $f(t, r) = 1$.
- *Transitivity:* For any records r, t, and u in R, if $f(r, t) = f(t, u) = 1$ then $f(r, u) = 1$.

The existing deep ER solutions, such as `DITTO`, disregard the equivalence constraints. I integrate these constraints with training deep ER models, and I conduct experiments in Section 6.2.3 to evaluate the impact of considering these constraints on the quality of record matching in `DITTO`.

Note that the partitioning of records also leads to other types of constraints. For example, one can infer a constraint that says if $f(r, t) = 0$ then $f(t, r) = 0$. Similarly, if $f(r, t) = 1$ and $f(t, u) = 0$, one can infer that $f(r, u) = 0$. This is studied in the context of conventional ER solutions in [6]. I postpone the study of these constraints in deep ER solutions to my future research.

4.3 Fairness Constraints

To formalize fairness constraints, I start by defining the AUC of an ER solution, and I introduce subAUC, an extension of the AUC for subpopulations.

The existing ER solutions often provide a binary classifier f that consists of a score function $s : R \times R \mapsto [0, 1]$ and a classification threshold $\theta \in [0, 1]$. The score function s receives a record pair and returns a real number that reflects the probability of matching the records. The class of a record-pair is decided using the threshold: $f(r_i, t_i) = 1$ if $\theta \leq s(r_i, t_i)$, and $f(r_i, t_i) = 0$ otherwise.

For an ER solution with a score function s and a threshold θ, the AUC measures the quality of s independent of the threshold θ. To formally define the AUC, I assume a probability distribution "Pr" with the following random variables: random variables X and Y that represent random records from R, a binary random variable L that is 1 if X and Y are equivalent and 0 otherwise, and a random variable S that takes values in $[0, 1]$ and represents the score of the random pair X and Y returned from the score function s. True positive rate (TPR) and false positive rate (FPR) for a classifier f (with s and θ) are defined as follows:

$$TPR_s(\theta) = Pr(S \geq \theta | L = 1) \text{ and } FPR_s(\theta) = Pr(S \geq \theta | L = 0). \tag{4.1}$$

The following equation defines the AUC of s using TPR and FPR [49]:

$$AUC_s = \int_0^1 TPR_s(FPR_s^{-1}(x))\,dx. \tag{4.2}$$

In Equation 4.2, FPR_s^{-1} is the inverse of FPR, and takes an input rate in $[0, 1]$ and returns the threshold θ for s that gives the input rate as FPR.

With fairness constraints, I seek to find an ER classifier that performs well in all subpopulations. I define subAUC, which measures the performance of a classifier in subpopulations. To do this, I assume records have a sensitive feature A, such as gender or race, with a discrete domain $\{a_1, ..., a_k\}$. I assume one sensitive feature for easier presentation, but this trivially extends to multiple sensitive features with discrete domains. I define performance measures TPR and FPR for a subpopulation $R_{A,a}$:

$$TPR_s^a(\theta) = Pr(S \geq \theta | L = 1, X.A = a \text{ or } Y.A = a), \tag{4.3}$$

$$FPR_s^a(\theta) = Pr(S \geq \theta | L = 0, X.A = a \text{ or } Y.A = a). \tag{4.4}$$

Using TPR and FPR for a subpopulation, I define subAUC as follows:

$$AUC_s^a = w^+ \times AUC_s^{a+}\,dx + w^- \times AUC_s^{a-}, \tag{4.5}$$

It is the probability that a pair of equivalent records is ranked higher than a pair of non-equivalent records when a record from $R_{A,a}$ is in either pair. In other words, subAUC considers any miss-ranked record pairs with at least one record from the subpopulation. This miss-ranked pair can cause a record from the subpopulation to be incorrectly merged with records from the same or other populations or the opportunity to merge it with an equivalent record might be lost as a result of this miss-ranking.

In Equation 4.5, AUC_s^{a+} and AUC_s^{a-} are also two probabilities defines as

$$AUC_s^{a+} = \int_0^1 TPR_s((FPR_s^a)^{-1}(x))\,dx \text{ and } AUC_s^{a-} = \int_0^1 TPR_s^a((FPR_s)^{-1}(x))\,dx \tag{4.6}$$

Here, AUC_s^{a+} is the probability that an equivalent pair in which at least one record is from $R_{A,a}$ is ranked higher than a nonequivalent pair. Similarly, AUC_s^{a-} is the probability that an equivalent pair is ranked higher than a nonequivalent pair in which at least one record is from $R_{A,a}$. To define the weights w^+ and w^- in Equation 4.5, I use a few new terms: $\mathcal{I} = R \times R$ is the set of all record pairs, $\mathcal{I}^+$ is the set of all possible equivalent pairs, and $\mathcal{I}_a = (R_{A,a} \times R) \cup (R \times R_{A,a})$ is the set of all possible record pairs with at least one record from the subpopulation $R_{A,a}$. I also use $\mathcal{I}_a^+ = \mathcal{I}_a \cap \mathcal{I}^+$ to refer to the set of all equivalent record pairs with at least one record from the subpopulation $R_{A,a}$. $\mathcal{I}^-$ and $\mathcal{I}_a^-$ are defined similarly. Using these terms, I define the weights w^+ and w^- as follows:

$$w^+ = \frac{|\mathcal{I}_a^+| \times |\mathcal{I}^-|}{|\mathcal{I}_a^+| \times |\mathcal{I}^-| + |\mathcal{I}^+| \times |\mathcal{I}_a^-|} \qquad w^- = \frac{|\mathcal{I}^+| \times |\mathcal{I}_a^-|}{|\mathcal{I}_a^+| \times |\mathcal{I}^-| + |\mathcal{I}^+| \times |\mathcal{I}_a^-|} \tag{4.7}$$

Intuitively, w^+ and w^- are, respectively, the number of comparisons between record pairs to compute AUC_s^{a+} and AUC_s^{a-}, respectively, i.e., $|\mathcal{I}_a^+| \times |\mathcal{I}^-|$ and $|\mathcal{I}^+| \times |\mathcal{I}_a^-|$, normalized by the total

number of comparisons required to compute these two probabilities, i.e., $|\mathcal{I}_a^+|\times|\mathcal{I}^-|+|\mathcal{I}^+|\times|\mathcal{I}_a^-|$. Note that both w^+ and w^- only depend on R, not s.

Now that I formally specified the notion of subAUC, I use it to define the bias of a score function. The bias is the gap between the subAUCs between the subpopulations:

$$Bias(s) = AUC_s^{\max} - AUC_s^{\min} \tag{4.8}$$

where $AUC_s^{\max}$ and $AUC_s^{\min}$ are the maximum and minimum subAUCs between the subpopulations. In the experiments, I use a normalized version of AUC as defined in [39] that allows me to compare bias in different experiments:

$$NormBias(s) = 1 - \frac{AUC_s^{\min}}{AUC_s^{\max}} \tag{4.9}$$

The normalized bias is in $[0, 1]$; it is 0 when the subpopulations have the subAUC, and it is 1 when the gap between the subAUCs is 1 (s correctly ranks all record pairs when records from one subpopulation are involved, and incorrectly ranks all the pairs with records from the other subpopulation). Note that, here I assume the maximum subAUC is always positive which is a reasonable assumption since even a random score function achieves the AUC equal to 0.5. A fairness constraint expects a score function s to have a bias lower than a bias threshold $\beta \in [0, 1]$, i.e., $NormBias(s) \leq \beta$. In the experiments in Chapter 6, I use $NormBias(s, D_T)$ to emphasize that the bias is computed using the dataset D_T.

The experimental results in Section 6.2.4 I measure the bias of `DITTO`, as one of the SOTA deep ER solutions, when it is applied with three different datasets. In Section 5.3, I present an algorithm for mitigating bias in `DITTO` by applying to training data.

Chapter 5

Incorporating Constraints with Entity Resolution

My solution for incorporating fairness and equivalence constraints with ER is an algorithm that uses DA (see Section 2.1 for a review of pre-, in-, and post-processing for integrating fairness constraints with ML models). I start by explaining the pre-processing for integrating equivalence constraints, which I call *data completion*, in Section 5.1, and then explain the DA for applying fairness constraints in Section 5.2. I present the complete algorithm in Section 5.3.

5.1 Data Completion for Equivalence Constraints

To incorporate equivalence constraints with a deep ER solution, I update the training data for the deep model by adding record pairs that are missing due to the violation of the equivalence constraints. I refer to this as *data completion*. Given a training dataset D, its data completion w.r.t. reflexivity means if there is a labeled record pair (r, t, l) in the training dataset D, two new record-pairs $(r, r, 1)$ and $(t, t, 1)$ must be added to D, if they are not already in D. Similarly, D's data completion w.r.t. symmetry means if there is a labeled record-pair $(r, t, 1)$ in D but not $(t, r, 1)$, the latter must be added. And, finally the data completion w.r.t. transitivity means if the labeled record-pairs $(r, t, 1)$ and $(t, u, 1)$ are in D, $(r, u, 1)$ must be added to D if it is not already in D.

In the experiments in Section 6.2.4, I analyze the impact of data completion on `DITTO` and show that it can improve the satisfaction of the equivalence constraints and additionally improve `DITTO` by decreasing bias.

5.2 Data Augmentation for Fairness Constraints

To mitigate bias, I looked into the sources of bias and a decreased subAUC in subpopulations. Bias in a subpopulation is either caused by low scores for equivalent pairs involving individual records from the subpopulation or high scores for nonequivalent pairs. These, in turn, can be caused by the lack of training samples in the form of equivalent or nonequivalent pairs. The lack of enough training samples of both types often happens in minority subpopulations with fewer overall training samples.

To mitigate biases in ER solutions, I apply DA, a common technique in computer vision and NLP that augments training data with new samples to improve the accuracy of ML models that use the training dataset. DA is also applied to improve the data preparation and cleaning processes [73, 78]. Note that DA has some disadvantages. While DA for one subpopulation improves its subAUC, it might reduce the subAUC of some other subpopulations. So, there is a trade-off between improving the subAUC of subpopulation and mitigating the overall AUC.

My solution to mitigate bias in ER and satisfy fairness constraints is to augment the training dataset with more labeled pairs from subpopulations with minimum subAUC. The DA has three main steps that are iteratively applied for adding new labeled pairs.

1. I first use the training data to find the score function s

2. I then compute subAUCs for s using the validation data and find the subpopulations with min and max subAUC.

3. Then, I use `Rotom` to create new records in the subpopulations with min subAUC by perturbing values in the current record pairs. After augmenting the training data, I continue with a new iteration and training in (I)

For computing AUCs (and subAUCs) given in a dataset D, I use the normalized Mann-Whitney U-Statistic that I denote by $\widehat{AUC}_s$ and provides an estimation of the AUC:

$$\widehat{AUC}_s = \frac{\sum_{p^+ \in D^+} \sum_{p^- \in D^-} \mathbb{1}_{s(p^+) \geq s(p^-)}}{|D^+| \times |D^-|} \tag{5.1}$$

In this estimation, D^+ and D^- are respectively the subsets of equivalent and nonequivalent pairs in D, and $\mathbb{1}_{s(p_i^+) \geq s(p_i^-)}$ is an indicator function that returns 1 if an equivalent pair p^+ is ranked higher than an non-equivalent pair p^-, and 0 otherwise. To compute the confidence interval of this estimation, I use bootstrapping [47].

I also estimate AUC_s^{a+} and AUC_s^{a+} using U-Statistic:

$$\widehat{AUC}_s^{a+} = \frac{\sum_{p^+ \in D_a^+} \sum_{p^- \in D^-} \mathbb{1}_{s(p^+) \geq s(p^-)}}{|D_a^+| \times |D^-|} \qquad \widehat{AUC}_s^{a-} = \frac{\sum_{p^+ \in D^+} \sum_{p^- \in D_a^-} \mathbb{1}_{s(p^+) \geq s(p^-)}}{|D^+| \times |D_a^-|} \tag{5.2}$$

The estimation of AUC_s^a is based on a similar formula:

$$\widehat{AUC}_s^a = \frac{\sum_{p^+ \in D_a^+} \sum_{p^- \in D^-} \mathbb{1}_{s(p^+) \geq s(p^-)} + \sum_{p^+ \in D^+} \sum_{p^- \in D_a^-} \mathbb{1}_{s(p^+) \geq s(p^-)}}{|D_a^+| \times |D^-| + |D^+| \times |D_a^-|} \tag{5.3}$$

In these estimations, D_a is the subset of record pairs in D with at least one pair in $R_{A,a}$. D_a^+ and D_a^- are defined similar to D^+ and D^-.

5.3 Debiasing Algorithm

Algorithm 1 shows the main steps of the algorithm, *Debiasing*, for incorporating equivalence and fairness constraints with training a score function for ER. The algorithm inputs a training dataset D_T, a validation dataset D_V, a sensitive feature A to specify subpopulations, a bias threshold β, and a tuning parameter K. The output is a score function s training by incorporating the constraints.

The algorithm starts in Line 1 by initializing S with the sensitive values that appear in either training or validation data, representing the subpopulations with these sensitive values. S_{aug}, which is initialized with $\emptyset$ in Line 2, is the set of subpopulations that the data augmentations are already applied for during the algorithm and is used by the algorithm to avoid reapplying data augmentation for a subpopulation. My experiments show that reapplying data augmentation does not improve performance in subpopulations. Lines 3 and 4 complete the training and test data to integrate equivalence constraints. The algorithm trains an initial score function s (Line 5), and iteratively applies DA (Line 8) for the subpopulations in S_{min} with lowest performance (AUC). The algorithm finds these subpopulations in Line 7, where the procedure *FindMinAUCs* computes the AUCs for all subpopulations and returns the bottom K subpopulations that are not in S_{aug} for which data augmentation is already applied. Note that computing AUCs is done using the validation data. s is retrained (fine-tune) using the augmented data D_Δ, and add the subpopulations S_{min}, which are considered in the current iteration, to S_{aug}. The iterations continue until the bias is reduced to the acceptable threshold β or all the subpopulations are considered. The algorithm returns the latest trained function s in Line 11.

Algorithm 1: *Debiasing*(D_T, D_V, β, K)

Input: A training Dataset D_T, a validation dataset D_V, a bias threshold β, and the number of subpopulations K

Output: A fair score function s.

1 $S \leftarrow SensitiveValues(D_T \cup D_V)$;
2 $S_{aug} \leftarrow \emptyset$;
3 $D_T \leftarrow CompleteData(D_T)$;
4 $D_V \leftarrow CompleteData(D_V)$
5 $s \leftarrow Train(D_T)$;
6 **while** $NormBias(s, D_V) > \beta$ **and** $S_{aug} \subset S$ **do**
7 $\quad S_{min} \leftarrow FindMinAUCs(s, D_V, K, S_{aug}))$;
8 $\quad D_\Delta \leftarrow AugmentData(D_T, S_{min})$;
9 $\quad s \leftarrow Retrain(s, D_\Delta)$;
10 $\quad S_{aug} \leftarrow S_{aug} \cup S_{min}$;
11 **return** s;

Chapter 6

Experiments

The experiments in this chapter have the following objectives:

1. The experiments in Section 6.2.1 evaluate the performance and bias of DITTO when trained on dirty data. The goal is to study the impact of data quality on performance and bias in ER.

2. The experiments in Section 6.2.2 establish the relevance of the equivalence constraints (reflexivity, symmetry, and transitivity) in ER with real-world datasets. More precisely, they show equivalence classes consisting of more than two records frequently appearing in these datasets.

3. The experiments in Section 6.2.3 show the impact of equivalence constraints on DITTO. These experiments look at the performance of DITTO when trained on data completed w.r.t. the equivalence constraints.

4. The experiments in Section 6.2.4 show the effect of the tuning parameter K on the algorithm and bias and accuracy in its iterations. The experiments also prove the algorithm's effectiveness in mitigating bias in unseen test data.

I start this chapter by explaining the experimental setting and the datasets used in the experiments. I discuss the experimental results and takeaways in Section 6.3.

6.1 Experimental Setup

I implemented the algorithm with Python 3.7. For data completion w.r.t. the equivalence constraints, I use *NetworkX* [1], a Python library for working with graphs. I also use *Ethnicolor* [2], a library for predicting ethnicity based on first name and last name. The other library that I use is *Gender-Guesser* [3]. This library predicts gender based on the first name. The two last libraries are used to add the required sensitive features to the datasets where these features are missing.

[1] https://networkx.org/

[2] https://pypi.org/project/ethnicolr/

[3] https://pypi.org/project/gender-guesser/

The training process runs a fixed number of epochs (10, 15, or 40 depending on the dataset). I conducted all experiments on *ComputeCanada* [4] with 4 GPUs, 6 tasks running on 4 nodes. I used `DITTO`'s implementation [5] and `Rotom`'s implementation [6], which are open-source on GitHub. I present the experiment results on three different datasets I explain in Section 6.1.1.

6.1.1 Datasets

I use three datasets from the ER benchmark in [65] with the statistics detailed in Table 6.1. These are popular datasets for evaluating ER systems, e.g., Magellan [64], DeepMatcher [81], and `DITTO` [71]. Each dataset consists of two structured tables of entity records of the same schema. The number of attributes ranges from 4 to 15. Each dataset is randomly split into training, test, and validation sets with a ratio of 3:1:1. In the following, I briefly explain these three datasets and the specific data preparations for the experiments in this thesis.

Dataset	Records	Split	Without Equiv. Const.		With Equiv. Const.	
			# Pairs	% POS	# Pairs	% POS
DBLP-GS	23,752	Train	17,223	0.2288	23752	2.2176
	9,249	Validation	5,742	0.2290	7,754	0.6596
	5,600	Test	5,742	0.2290	–	–
Amazon-Google	2,853	Train	6,874	0.1121	6947	0.1250
	1,838	Validation	2,292	0.1121	2,300	0.1175
	2,059	Test	2,293	0.1121	–	–
FEBRL	4,744	Train	2,400	0.3333	2492	0.3844
	1,422	Validation	800	0.3333	1422	0.3683
	1,420	Test	800	0.3333	–	–

Table 6.1: Datasets tatistics

DBLP-GoogleScholar (GS): This dataset contains data about papers and their authors. Each record has five different attributes: title, author, ethnicity, etc. The dataset is generated in the Magellan project [63], which was initially used as a benchmark to study the system developed in the project. I use ethnicity as the sensitive attribute for defining subpopulations. The records do not have ethnicity as they refer to papers. I added ethnicity by finding the ethnicity of the majority of the authors or the first author if there is a tie. The ethnicity attribute defines 13 subpopulations.

Amazon-Google: This dataset is about products from Amazon and Google and has three attributes: title, manufacturer, and price. Similar to DBLP-ACM, this dataset is also generated in the Magellan project [63]. I use the manufacturer feature as the sensitive attribute to study

[4] https://www.computecanada.ca/
[5] https://github.com/megagonlabs/ditto
[6] https://github.com/megagonlabs/rotom

the quality of record matching for products from different manufacturers. In total, there are 283 subpopulations for the manufacturer attribute.

FEBRL: The FEBRL [7](freely extensible biomedical record linkage) the dataset is a comparison of patterns which is obtained in an epidemiological cancer study in Germany. The comparison patterns were created by the Institute for Medical Biostatistics, Epidemiology, and Informatics (IMBEI). The dataset is available for research online. Four datasets were generated from patient records, and I combined them to make pairs and feed the model. FEBRL dataset consists of fifteen attributes, such as given name, address, and postal code. The sensitive attribute is ethnicity which I added with ethnicolor library.

Table 6.1 (the left side of the vertical bar) shows the number of unique records for train, validation, and test dataset separately. It also shows the number of record pairs and the percentage of equivalent pairs (with label 1).

6.2 Experimental Results

6.2.1 Impact of Data Quality on Performance and Bias

This section aims to see how the performance and bias in DITTO change for varying levels of dirty data. The negative impact of low quality on ER is already known [71]. Here, I designed an experiment to systematically study performance and bias for varying levels of dirty data. There are different types of dirty data, such as outdated data, incomplete data, incorrect/inaccurate data, and inconsistent data. In this experiment, I consider incomplete data as missing values in the datasets. Other types of dirty data can be considered with ER that I postpone to my future work.

To this end, by randomly introducing missing values, I applied three levels of data dirtiness to the training and validation dataset on DBLP-GS and FEBRL datasets. Note that DBLP-GS already has almost 10% missing values. I add 10% more missing values in each step. Table 6.2 contains the detail of the results that shows DITTO is resistant to data dirtiness, as precision, recall, and F1 measure only slightly change with more missing values. Furthermore, the results show that the bias significantly increased when I increased the missing values. In the next experiments, I use DBLP-GS with 40% missing values to highlight the result of the algorithm in reducing bias.

I repeated the experiment with FEBRL, where I started from 30% for adding missing values to the original dataset. The performance did not change when I applied 30% missing values, which confirms DITTO's high resistance to dirty data. To see the breaking point, I added 10% more (40% missing values in total) and reported the performance changes in Table 6.2.

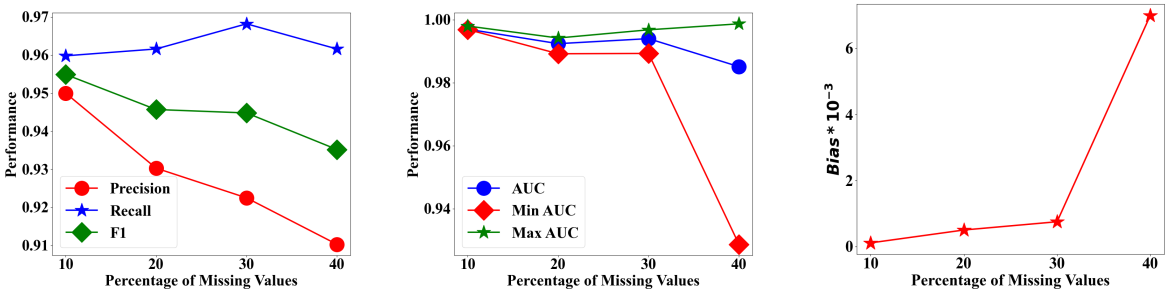
Figure 6.1 shows the results of the same experiment using Amazon-Google. It is clear from Figure 6.1a that F1 and precision decreased consistently by adding missing values in each step. Still, the recall has not decreased because missing values do not change the percentage of matched record pairs in training, which directly impacts recall. In Figure 6.1b, the Min AUC on the 40% missing value dropped significantly, but AUC has not decreased that much since

[7]https://github.com/J535D165/FEBRL-fork-v0.4.2

there are only a few subpopulations with significantly low AUCs. Figure 6.1c shows that bias has a sharp increase on 40% missing values, which shows one or more subpopulations with Min AUC have significantly been impacted by the missing values.

Missing (~ %)	Precision	Recall	F1	AUC	Min AUC	Max AUC	Bias
10 (default)	0.9500	0.9598	0.9549	0.9970	0.9969	0.9980	0.0011
20	0.9303	0.9616	0.9457	0.9924	0.9892	0.9942	0.0050
30	0.9225	0.9682	0.9448	0.9940	0.9893	0.9968	0.0075
40	0.9103	0.9616	0.9352	0.9851	0.9287	0.9987	0.0700

Table 6.2: DITTO's performance trained using DBLP-GS with varying missing values



(a) (b) (c)

Figure 6.1: DITTO's performance trained using Amazon-Google with varying missing values

6.2.2 Relevance of Equivalence Constraints

To study the impact of equivalence constraints on DITTO, I tried two scenarios. In the first scenario, I applied the equivalence constraints on training, validation, and test. In this scenario, the performance of DITTO, in terms of precision, recall, and F1, significantly improved, which was expected. In the second scenario, I only applied the equivalence constraints on the training and validation datasets while keeping the test data unchanged. This scenario does not evaluate the satisfaction of the constraints in the test dataset since the test does not include missing pairs for the constraints. The experiments in this scenario aim to see the impact of these constraints on a fixed test dataset and have a fair comparison with DITTO without the constraints. In this section, I report statistics about the added record pairs to the training and validation datasets, and I report the main results in the next section.

Analyzing the datasets shows that reflexivity and symmetry are ignored, and there are missing record pairs as violations of these constraints. For transitivity, I show in Figure 6.2 the number of equivalent classes in these datasets with more than 2 records. This shows transitive closure and completing the data w.r.t. transitivity can create new data, which is confirmed in Figure 6.3. This figure shows the number of new record pairs to be added to complete the data w.r.t. transitivity.

Table 6.1 (the right side of the table) shows detailed stats about the datasets after data completion w.r.t. the equivalence constraints. By applying symmetry on Amazon-Google, 6,874 pairs have been added to the dataset since non of the pairs did not satisfy symmetry; the number of added pairs is equal to the original size of pairs. For reflexivity, I added 13,748 pairs to the dataset similarly; since non of the pairs did not satisfy reflexivity, I had to add the double size of pairs to the original data set.

Figure 6.3c shows that record pairs have been added consistently until entity 21; the significant difference in entity 45 shows that 45 entities have a lot of missing records in terms of transitivity, so 900 records should be added to complete the graph and satisfy transitivity. Figure 6.2c proves this claim; it shows there is only one class that has 45 nodes or entities. In Figure 6.3, the number of edges that have been added starts from three nodes since transitivity is meaningful for at least three records.

6.2.3 Impact of Equivalence Constraints

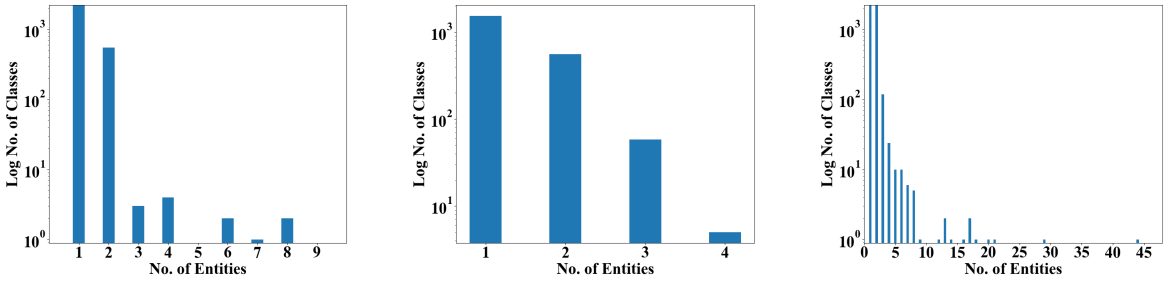
Tables 6.3 and 6.4 show the performance of DITTO with and without the equivalence constraints. Table 6.3 report the results using Amazon-Google, where the first row shows the performance without any constraints, and the second and the third row displays the results for DITTO with symmetry and with reflexivity. The results in the table show that there is no significant difference between DITTO with and without applying these two constraints. For symmetry, this can be because the input of DITTO is a string where the order of the records in the string does not impact the model. For reflexivity, the similarity between a record and itself is already learned by the model, and adding the new pairs that represent reflexivity does not add more knowledge. For transitivity, I conduct a separate experiment using all three datasets. In this experiment, I compared the performance of DITTO with and without transitive closure. The results in Table 6.4 improved recall while precision and F1 are decreased. This can be because of the fact that the training data is augmented with new equivalent pairs (with label 1). F1 and precision are reduced because the test dataset is balanced with more pairs labeled 0. For all three constraints, the interesting result is the increase in bias after applying the constraints.

	Precision	Recall	F1	AUC	Min AUC	Max AUC	Bias
DITTO	0.666	0.811	0.732	0.966	0.325	1.000	0.688
+Symm	0.665	0.812	0.731	0.955	0.341	1.000	0.659
+Refl	0.659	0.820	0.730	0.950	0.349	1.000	0.651

Table 6.3: DITTO's performance with and without the equivalence constraints

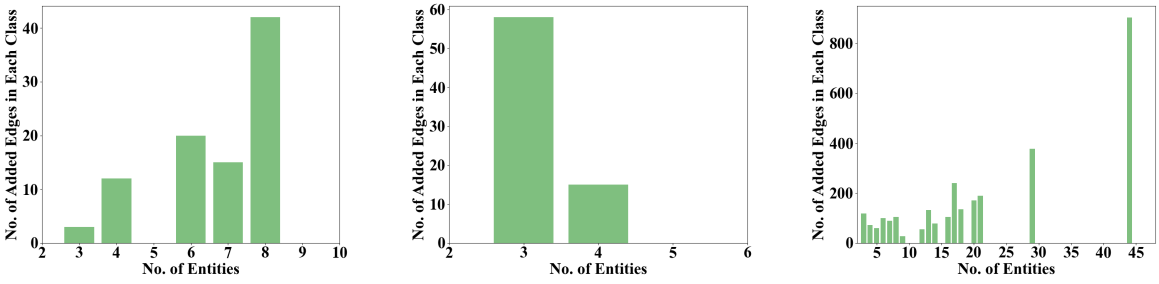
Dataset	Without Transitive Closure				With Transitive Closure			
	Precision	Recall	F1	AUC	Precision	Recall	F1	AUC
DBLP-GS	0.950	0.959	0.954	0.996	0.852(-0.097)	0.994(+0.035)	0.9180(-0.036)	0.9972(+0.0003)
Amazon-Google	0.666	0.812	0.732	0.966	0.651(-0.163)	0.829(+0.083)	0.729(-0.003)	0.968(+0.002)
FEBRL	0.995	0.992	0.995	0.991	0.993(-0.002)	0.994(+0.002)	0.997(+0.002)	0.993(+0.002)

Table 6.4: DITTO's performance with and without the equivalence constraints



(a) Patients (b) Amazon-Google (c) DBLP-GS

Figure 6.2: The number of equivalence classes vs class size



(a) FEBRL (b) Amazon-Google (c) DBLP-GS

Figure 6.3: The number of added edges (records) in each class

6.2.4 Analysis of Debiasing Algorithm

The first experiment in this section looks at the role of the tuning parameter K in the algorithm. This experiment uses Amazon-Google. The results in Figure 6.4 show that for large values of K, e.g., $K = 2$ and 3, there is no trend in bias as it fluctuates for varying iterations. For $K = 1$, the bias consistently decreases in each iteration. However, $K = 2$ and 3 are more efficient in runtime as more subpopulations are treated in each iteration. This means there is a trade-off between runtime performance of consistency in reducing bias for different values of K. In the other experiments, I use $K = 1$ to obtain consistent results.

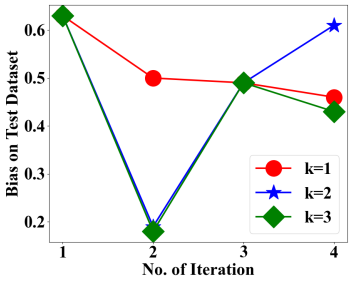
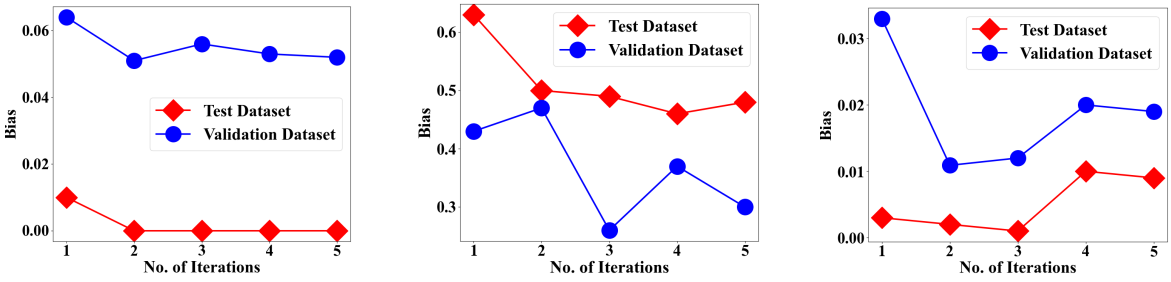


Figure 6.4: The bias in the algorithm for varying K

Using $K = 1$, I run the algorithm for the three datasets. Figure 6.5 shows how bias changes

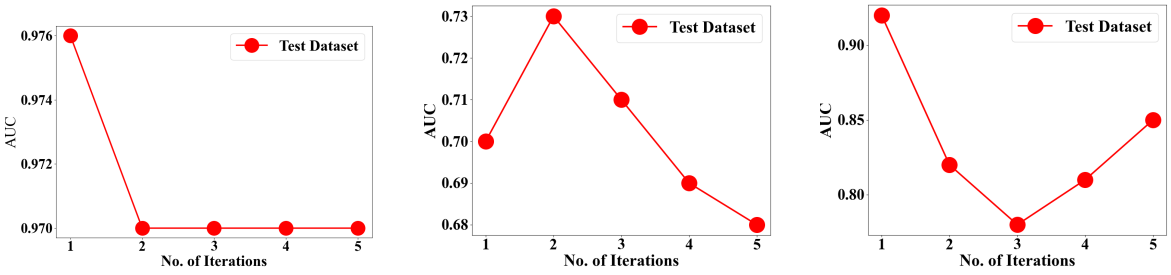
in the validation and test data in 5 iterations. The figure generally shows improvement in bias for validation and test dataset over five iterations on the *ethnicity* attribute. Figure 6.5b shows that bias has decreased consistently on validation and test dataset in the *manufacturer* attribute. The accuracy of ER for certain populations with few record pairs in Amazon-Google is lower than a random classifier, and that is the reason the bias on the test dataset is high (e.g., 0.6) in Figure 6.5b. In figure 6.5c, the bias has increased after iteration 4 on validation and test dataset on the *ethnicity* attribute. In this dataset, the number of each subpopulation was large. Applying DA to each subpopulation adds significant record pairs that change model performance for other subpopulations and increase bias.

Figure 6.6 shows the overall AUC on the test dataset where the AUC decreases in the debiasing algorithm as expected. This underlines the trade-off between bias and AUC. When we compare this figure with Figure 6.5 we can see the opposite behavior on the bias.



(a) FEBRL (b) Amazon-Google (c) DBLP-GS

Figure 6.5: The algorithm's bias in each iteration



(a) FEBRL (b) Amazon-Google (c) DBLP-GS

Figure 6.6: The algorithm's AUC in each iteration on test dataset

6.3 Discussion

The main takeaway from the experiments in Section 6.2.1 is that `DITTO` is resistant to dirty data in the form of missing values as its overall accuracy slightly decreases with more missing values. However, it is more sensitive w.r.t. its bias meaning more missing values introduce

significant bias. This means data dirtiness has a different impact on subpopulations and can have a more severe negative effect on specific subpopulations.

The general message in the experiments in Section 6.2.2 is that these constraints can be used to add more data to an existing dataset. The experiments in Section 6.2.3 showed that incorporating equivalence constraints improve their satisfaction of them in DITTO, but it has an insignificant impact on the quality of DITTO w.r.t. a fixed test dataset. It also showed that incorporating these constraints can decrease bias in DITTO.

The last set of experiments in Section 6.2.4 demonstrates that the algorithm can reduce bias in test and validation if it is tuned. There is a trade-off between the amount of DA in each iteration in terms of the number of subpopulations DA is applied for vs. the consistency in reducing bias. Training a fair ER model takes around one hour on average, which is multiple times the time needed to train DITTO. This high cost of training is justified when fairness in record matching is an important requirement. Techniques such as fine-tuning and retraining can be used to reduce the time to run the debiasing algorithm and train a fair model, which I postpone to my future work.

Chapter 7

Conclusion and Future Work

I studied the problem of incorporating constraints with ER. I focused on two types of constraints: equivalence and fairness. The experimental results using three datasets show that data completion w.r.t. transitivity constraint as an equivalence constraint that applies transitivity of matched records has a positive impact on the performance of DITTO and improves its accuracy. The experiments also revealed that due to its architecture, data completion w.r.t. reflexivity and symmetry do not significantly impact the performance of DITTO. For fairness constraints, I started by formalizing the notion of bias based on AUC. The fairness constraints expect an equal risk of record matching between all subpopulations. I formalized this by defining AUC per subpopulation. I used data augmentation using the Rotom framework to reduce bias in DITTO. The experimental result proves the effectiveness of this data augmentation solution in reducing bias. This thesis can be extended in several future research directions.

1. In this thesis, I consider fairness and equivalence constraints. A future research direction is to study other constraints, such as aggregate constraints.

2. I defined bias using subAUC and the gap between the accuracy of record matching in subpopulations in terms of their subAUC. Other notions of bias can be used to formalize the bias of ER.

3. A possible future research idea is to use in-processing (e.g., integrating fairness constraints with ML models' loss function) or post-processing (e.g., using clustering techniques, such as correlation clustering.

4. I used DITTO with BERT as the LM in the model to run the experiments. One can investigate bias in DITTO with other LMs or in deep ER solutions other than DITTO.

5. In this thesis, I focused on structured relational data. Another possible research idea is to study bias in ER for heterogeneous, unstructured, or semi-structured data.

6. I studied data dirtiness in the form of missing values. A possible future research direction is to investigate the impact of other forms of dirty data, e.g., inconsistency or stale values, on performance and bias in ER.

7. The debiasing algorithm is costly as it requires training ER models many times while monitoring their bias and performance using the validation data. I intend to look into

ways to improve the performance of the debiasing algorithm by fine-tuning the trained models in each round instead of training a new model.

Bibliography

[1] Gao highlights need to better match patient records. https://www.pewtrusts.org/en/research-and-analysis/articles/2019/01/15/gao-highlights-need-to-better-match-patient-records, 2019. [Online; accessed 5-July-2022]. 2

[2] Four cooperative agreements: Census bureau research on record linkage and entity resolution. https://www.census.gov/newsroom/blogs/research-matters/2021/10/four-cooperative-agreements.html, 2021. [Online; accessed 5-July-2022]. 2

[3] Record linkage: Statistics canada. https://www150.statcan.gc.ca/n1/edu/power-pouvoir/ch3/5214780-eng.htm, 2021. [Online; accessed 5-July-2022]. 2

[4] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)*, 55(5):1–27, 2008. iii, 2, 3, 20

[5] Arvind Arasu, Christopher Ré, and Dan Suciu. Large-scale deduplication with constraints using dedupalog. In *2009 IEEE 25th International Conference on Data Engineering*, pages 952–963. IEEE, 2009. 12

[6] Jurian Baas, Mehdi M Dastani, and Ad J Feelders. Exploiting transitivity for entity matching. In *European Semantic Web Conference*, pages 109–114. Springer, 2021. 4, 20

[7] Jurian Baas, Mehdi M. Dastani, and Ad J. Feelders. Exploiting transitivity for entity matching. In *The Semantic Web: ESWC 2021 Satellite Events: Virtual Event, June 6–10, 2021, Revised Selected Papers*, page 109–114, 2021. 4

[8] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine learning*, 56(1):89–113, 2004. 3

[9] Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research*, 50(1):3–44, 2021. 5

[10] Mikhail Bilenko and Raymond J Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 39–48, 2003. 3

[11] Sumon Biswas and Hridesh Rajan. Fair preprocessing: Towards understanding compositional fairness of data transformers in machine learning pipeline. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, page 981–993, 2021. 8

[12] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, 2017. 3

[13] Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. Learning structured embeddings of knowledge bases. In *Twenty-fifth AAAI conference on artificial intelligence*, 2011. 2

[14] Ursin Brunner and Kurt Stockinger. Entity matching with transformer architectures-a step forward in data integration. In *International Conference on Extending Database Technology*, 2020. 3, 4

[15] Toon Calders and Sicco Verwer. Three naive bayes approaches for discrimination-free classification. *Data mining and knowledge discovery*, 21(2):277–292, 2010. 11

[16] Surajit Chaudhuri, Anish Das Sarma, Venkatesh Ganti, and Raghav Kaushik. Leveraging aggregate constraints for deduplication. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 437–448, 2007. 4, 11

[17] Zhaoqiang Chen, Qun Chen, Boyi Hou, Zhanhuai Li, and Guoliang Li. Towards interpretable and learnable risk analysis for entity resolution. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 1165–1180, 2020. 4

[18] Peter Christen. Febrl: a freely available record linkage system with a graphical user interfac. In *the second Australasian workshop on Health data and knowledge management*, volume 80, pages 14–25, 2008. 3

[19] Peter Christen. The data matching process. In *Data matching*, pages 23–35. Springer, 2012. 14

[20] Vassilis Christophides, Vasilis Efthymiou, and Kostas Stefanidis. Entity resolution in the web of data. *Synthesis Lectures on the Semantic Web*, 5(3):1–122, 2015. 2

[21] William W. Cohen and Jacob Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, page 475–480, 2002. 3

[22] Ricardo G Cota, Marcos André Gonçalves, and Alberto HF Laender. A heuristic-based hierarchical clustering method for author name disambiguation in digital libraries. In *SBBD*, pages 20–34. Citeseer, 2007. 3, 20

[23] Sanjib Das, Paul Suganthan GC, AnHai Doan, Jeffrey F Naughton, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, Vijay Raghavendra, and Youngchoon Park. Falcon: Scaling up hands-off crowdsourced entity matching to build cloud services. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1431–1446, 2017. 12

[24] Erik D Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2-3):172–187, 2006. 3

[25] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the 21st international conference on World Wide Web*, pages 469–478, 2012. 12

[26] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Large-scale linked data integration using probabilistic reasoning and crowdsourcing. *The VLDB Journal*, 22(5):665–687, 2013. 3

[27] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 14

[28] Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava. Integrating conflicting data: the role of source dependence. *Proceedings of the VLDB Endowment*, 2(1):550–561, 2009. iii, 2

[29] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012. 10

[30] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. Deeper–deep entity resolution. *arXiv preprint arXiv:1710.00597*, 2017. 3

[31] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. Distributed representations of tuples for entity resolution. *Proceedings of the VLDB Endowment*, 11(11):1454–1467, 2018. 3

[32] Vasilis Efthymiou, Kostas Stefanidis, and Vassilis Christophides. Big data entity resolution: From highly to somehow similar entity descriptions in the web. In *2015 IEEE International Conference on Big Data (Big Data)*, pages 401–410. IEEE, 2015. 2

[33] Vasilis Efthymiou, Kostas Stefanidis, Evaggelia Pitoura, and Vassilis Christophides. FairER: entity resolution with fairness constraints. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3004–3008, 2021. 4, 8, 12

[34] Wenfei Fan, Xibei Jia, Jianzhong Li, and Shuai Ma. Reasoning about record matching rules. *Proc. VLDB Endow.*, 2(1):407–418, 2009. 3

[35] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006. 6

[36] Ivan P Fellegi and Alan B Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969. 3

[37] Donatella Firmani, Sainyam Galhotra, Barna Saha, and Divesh Srivastava. Robust entity resolution using a crowdoracle. *IEEE Data Eng. Bull.*, 41(2):91–103, 2018. 3

[38] Jeffrey Fisher, Peter Christen, Qing Wang, and Erhard Rahm. A clustering-based framework to control block sizes for entity resolution. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 279–288, 2015. 3, 20

[39] Hortense Fong, Vineet Kumar, Anay Mehrotra, and Nisheeth K Vishnoi. Fairness for auc via feature augmentation. *arXiv preprint arXiv:2111.12823*, 2021. 5, 10, 22

[40] Cheng Fu, Xianpei Han, Jiaming He, and Le Sun 0001. Hierarchical matching network for heterogeneous entity resolution. In *IJCAI International Joint Conference in Artificial Intelligence*, pages 3665–3671, 2020. 3

[41] Cheng Fu, Xianpei Han, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. End-to-end multi-perspective matching for entity resolution. In *IJCAI International Joint Conference in Artificial Intelligence*, 2019. 3

[42] Jim Gemmell, Benjamin IP Rubinstein, and Ashok K Chandra. Improving entity resolution with global constraints. *arXiv preprint arXiv:1108.6016*, 2011. 4, 12

[43] Lise Getoor and Ashwin Machanavajjhala. Entity resolution: theory, practice & open challenges. *Proceedings of the VLDB Endowment*, 5(12):2018–2019, 2012. 2

[44] Chaitanya Gokhale, Sanjib Das, AnHai Doan, Jeffrey F Naughton, Narasimhan Rampalli, Jude Shavlik, and Xiaojin Zhu. Corleone: Hands-off crowdsourcing for entity matching. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 601–612, 2014. 3

[45] Chaitanya Gokhale, Sanjib Das, AnHai Doan, Jeffrey F Naughton, Narasimhan Rampalli, Jude Shavlik, and Xiaojin Zhu. Corleone: Hands-off crowdsourcing for entity matching. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 601–612, 2014. 12

[46] Yash Govind, Erik Paulson, Palaniappan Nagarajan, Paul Suganthan GC, AnHai Doan, Youngchoon Park, Glenn Fung, Devin Conathan, Marshall Carter, and Mingju Sun. Cloudmatcher: a hands-off cloud/crowd service for entity matching. 2018. 3

[47] Jiezhun Gu, Subhashis Ghosal, and Anindya Roy. Bayesian bootstrap estimation of roc curve. *Statistics in medicine*, 27(26):5407–5420, 2008. 24

[48] Sara Hajian and Josep Domingo-Ferrer. A methodology for direct and indirect discrimination prevention in data mining. *IEEE transactions on knowledge and data engineering*, 25(7):1445–1459, 2012. 11

[49] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982. 21

[50] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016. 10, 11

[51] Vasileios Iosifidis and Eirini Ntoutsi. Dealing with bias via data augmentation in supervised learning scenarios. *Jo Bates Paul D. Clough Robert Jäschke*, 24, 2018. 11

[52] Di Jin, Bunyamin Sisman, Hao Wei, Xin Luna Dong, and Danai Koutra. Deep transfer learning for multi-source entity linkage via domain adaptation. *arXiv preprint arXiv:2110.14509*, 2021. 12

[53] Nathan Kallus and Angela Zhou. *The Fairness of Risk Scores beyond Classification: Bipartite Ranking and the XAUC Metric*. 2019. 5, 7, 10

[54] Faisal Kamiran and Toon Calders. Data preprocessing techniques for classification without discrimination. *Knowledge and information systems*, 33(1):1–33, 2012. 11

[55] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 35–50. Springer, 2012. 11

[56] Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. Fairness-aware learning through regularization approach. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 643–650. IEEE, 2011. 11

[57] Alexandros Karakasidis and Evaggelia Pitoura. Identifying bias in name matching tasks. In *EDBT International Conference on Extending Database Technology*, pages 626–629, 2019. 8

[58] Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. Low-resource deep entity resolution with transfer and active learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5851–5861, 2019. 3, 12

[59] Niki Kilbertus, Mateo Rojas Carulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. Avoiding discrimination through causal reasoning. *Advances in neural information processing systems*, 30, 2017. 11

[60] Michael P Kim, Amirata Ghorbani, and James Zou. Multiaccuracy: Black-box post-processing for fairness in classification. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 247–254, 2019. 11

[61] Nishadi Kirielle, Peter Christen, and Thilina Ranbaduge. Transer: Homogeneous transfer learning for entity resolution. In *EDBT International Conference on Extending Database Technology*, pages 2–118, 2022. 13

[62] Lars Kolb, Hanna Köpcke, Andreas Thor, and Erhard Rahm. Learning-based entity resolution with mapreduce. In *Proceedings of the third international workshop on Cloud data management*, pages 1–6, 2011. 3

[63] Pradap Konda, Sanjib Das, AnHai Doan, Adel Ardalan, Jeffrey R Ballard, Han Li, Fatemah Panahi, Haojun Zhang, Jeff Naughton, Shishir Prasad, et al. Magellan: toward building entity matching management systems over data science stacks. *Proceedings of the VLDB Endowment*, 9(13):1581–1584, 2016. 27

[64] Pradap Venkatramanan Konda. *Magellan: Toward building entity matching management systems*. The University of Wisconsin-Madison, 2018. 27

[65] Hanna Köpcke, Andreas Thor, and Erhard Rahm. Evaluation of entity resolution approaches on real-world match problems. *Proceedings of the VLDB Endowment*, 3(1-2):484–493, 2010. 27

[66] Pigi Kouki, Jay Pujara, Christopher Marcum, Laura Koehly, and Lise Getoor. Collective entity resolution in familial networks. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 227–236. IEEE, 2017. 4

[67] Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. *Advances in neural information processing systems*, 30, 2017. 10

[68] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195, 2015. 2

[69] Guoliang Li, Yudian Zheng, Ju Fan, Jiannan Wang, and Reynold Cheng. Crowdsourced data management: Overview and challenges. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1711–1716, 2017. 12

[70] Lingli Li, Jianzhong Li, and Hong Gao. Rule-based method for entity resolution. *IEEE Transactions on Knowledge and Data Engineering*, 27(1):250–263, 2014. 3

[71] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. Deep entity matching with pre-trained language models. *arXiv preprint arXiv:2004.00584*, 2020. vii, 3, 4, 14, 15, 27, 28

[72] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, Jin Wang, Wataru Hirota, and Wang-Chiew Tan. Deep entity matching: Challenges and opportunities. *J. Data and Information Quality*, 13(1), 2021. 3

[73] Yuliang Li, Xiaolan Wang, Zhengjie Miao, and Wang-Chiew Tan. Data augmentation for ml-driven data preparation and integration. *Proceedings of the VLDB Endowment*, 14(12):3182–3185, 2021. 17, 24

[74] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 4, 14

[75] Pranay K Lohia, Karthikeyan Natesan Ramamurthy, Manish Bhide, Diptikalyan Saha, Kush R Varshney, and Ruchir Puri. Bias mitigation post-processing for individual and group fairness. In *Icassp 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 2847–2851. IEEE, 2019. 11

[76] Lacramioara Mazilu, Norman W Paton, Nikolaos Konstantinou, and Alvaro AA Fernandes. Fairness in data wrangling. In *2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)*, pages 341–348. IEEE, 2020. 8

[77] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35, 2021. 5, 10

[78] Zhengjie Miao, Yuliang Li, and Xiaolan Wang. Rotom: A meta-learned data augmentation framework for entity matching, data cleaning, text classification, and beyond. In *Proceedings of the 2021 International Conference on Management of Data*, SIGMOD '21, page 1303–1316, 2021. vii, viii, 8, 14, 16, 18, 24

[79] Shubhanshu Mishra, Sijun He, and Luca Belli. Assessing demographic bias in named entity recognition. *arXiv preprint arXiv:2008.03415*, 2020. 8

[80] Alvaro Monge and Charles Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. 1997. 3

[81] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 International Conference on Management of Data*, pages 19–34, 2018. 3, 27

[82] Razieh Nabi and Ilya Shpitser. Fair inference on outcomes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 11

[83] Youcef Nafa, Qun Chen, Zhaoqiang Chen, Xingyu Lu, Haiyang He, Tianyi Duan, and Zhanhuai Li. Active deep learning on entity resolution by risk sampling. *Knowledge-Based Systems*, 236:107729, 2022. 3

[84] Howard B Newcombe, James M Kennedy, SJ Axford, and Allison P James. Automatic linkage of vital records: Computers can be used to extract" follow-up" statistics of families from files of routine records. *Science*, 130(3381):954–959, 1959. 3

[85] Hao Nie, Xianpei Han, Ben He, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. Deep sequence-to-sequence entity matching for heterogeneous entity resolution. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 629–638, 2019. 3

[86] Matteo Paganelli, Paolo Sottovia, Francesco Guerra, and Yannis Velegrakis. Tuner: Fine tuning of rule-based entity matchers. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2945–2948, 2019. 3

[87] Fatemah Panahi, Wentao Wu, AnHai Doan, and Jeffrey F Naughton. Towards interactive debugging of rule-based entity matching. In *EDBT International Conference on Extending Database Technology*, pages 354–365, 2017. 3

[88] George Papadakis, Ekaterini Ioannou, and Themis Palpanas. Entity resolution: Past, present and yet-to-come. In *EDBT International Conference on Extending Database Technology*, pages 647–650, 2020. 2, 20

[89] George Papadakis, Ekaterini Ioannou, Emanouil Thanos, and Themis Palpanas. The four generations of entity resolution. *Synthesis Lectures on Data Management*, 16(2):1–170, 2021. 2

[90] Hanna Pasula, Bhaskara Marthi, Brian Milch, Stuart J Russell, and Ilya Shpitser. Identity uncertainty and citation matching. *Advances in neural information processing systems*, 15, 2002. 3

[91] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 3

[92] Dana Pessach and Erez Shmueli. Algorithmic fairness. *arXiv preprint arXiv:2001.09784*, 2020. 5

[93] Felix Petersen, Debarghya Mukherjee, Yuekai Sun, and Mikhail Yurochkin. Post-processing for individual fairness. *Advances in Neural Information Processing Systems*, 34:25944–25955, 2021. 11

[94] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. *Advances in neural information processing systems*, 30, 2017. 11

[95] Anna Primpeli, Christian Bizer, and Margret Keuper. Unsupervised bootstrapping of active learning for entity resolution. In *European Semantic Web Conference*, pages 215–231. Springer, 2020. 12

[96] Kun Qian, Lucian Popa, and Prithviraj Sen. Active learning for large-scale entity resolution. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 1379–1388, 2017. 3

[97] Kun Qian, Lucian Popa, and Prithviraj Sen. Active learning for large-scale entity resolution. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1379–1388, 2017. 12

[98] Inioluwa Deborah Raji and Joy Buolamwini. Actionable auditing: Investigating the impact of publicly naming biased performance results of commercial ai products. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 429–435, 2019. 5

[99] Babak Salimi, Luke Rodriguez, Bill Howe, and Dan Suciu. Interventional fairness: Causal database repair for algorithmic fairness. In *Proceedings of the 2019 International Conference on Management of Data*, pages 793–810, 2019. 11

[100] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 4, 14

[101] Sebastian Schelter, Yuxuan He, Jatin Khilnani, and Julia Stoyanovich. Fairprep: Promoting data to a first-class citizen in studies on fairness-enhancing interventions. *arXiv preprint arXiv:1911.12587*, 2019. 8

[102] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. Recommendations as treatments: Debiasing learning and evaluation. In *international conference on machine learning*, pages 1670–1679. PMLR, 2016. 5

[103] Shubham Sharma, Yunfeng Zhang, Jesús M Ríos Aliaga, Djallel Bouneffouf, Vinod Muthusamy, and Kush R Varshney. Data augmentation for discrimination prevention and bias disambiguation. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 358–364, 2020. 11

[104] Warren Shen, Xin Li, and AnHai Doan. Constraint-based entity matching. In *AAAI*, pages 862–867, 2005. 4, 12

[105] Rohit Singh, Vamsi Meduri, Ahmed Elmagarmid, Samuel Madden, Paolo Papotti, Jorge-Arnulfo Quiané-Ruiz, Armando Solar-Lezama, and Nan Tang. Generating concise entity matching rules. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, page 1635–1638, 2017. 3

[106] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014. 17

[107] Saravanan Thirumuruganathan, Shameem A Puthiya Parambath, Mourad Ouzzani, Nan Tang, and Shafiq Joty. Reuse and adaptation for entity resolution through transfer learning. *arXiv preprint arXiv:1809.11084*, 2018. 13

[108] Vasilis Verroios and Hector Garcia-Molina. Entity resolution with crowd errors. In *2015 IEEE 31st International Conference on Data Engineering*, pages 219–230. IEEE, 2015. 3

[109] Vassilios S Verykios, George V Moustakides, and Mohamed G Elfeky. A bayesian decision model for cost optimal record matching. *The VLDB Journal*, 12(1):28–40, 2003. 3

[110] Jiannan Wang, Tim Kraska, Michael J. Franklin, and Jianhua Feng. Crowder: Crowdsourcing entity resolution. *Proc. VLDB Endow.*, 5(11):1483–1494, 2012. 3

[111] Jiannan Wang, Tim Kraska, Michael J Franklin, and Jianhua Feng. Crowder: Crowdsourcing entity resolution. *arXiv preprint arXiv:1208.1927*, 2012. 12

[112] Jiannan Wang, Guoliang Li, Jeffrey Xu Yu, and Jianhua Feng. Entity matching: How similar is similar. *Proc. VLDB Endow.*, 4(10):622–633, 2011. 3

[113] Renzhi Wu, Sanya Chaba, Saurabh Sawlani, Xu Chu, and Saravanan Thirumuruganathan. Zeroer: Entity resolution using zero labeled examples. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 1149–1164, 2020. 4

[114] Ke Yang, Biao Huang, Julia Stoyanovich, and Sebastian Schelter. Fairness-aware instrumentation of preprocessing pipelines for machine learning. In *Workshop on Human-In-the-Loop Data Analytics (HILDA'20)*, 2020. 8

[115] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019. 4

[116] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 335–340, 2018. 11

[117] Chen Zhao and Yeye He. Auto-em: End-to-end fuzzy entity-matching using pre-trained deep models and transfer learning. In *The World Wide Web Conference*, pages 2413–2424, 2019. 13

Curriculum Vitae

Name:	Soudeh Nilforoushan
Post-Secondary Education and Degrees:	Amirkabir University of Technology Tehran, Tehran, Iran 2015-2020 B.Sc. University of Western Ontario London, ON 2021 - 2022 MS.c.
Related Work Experience:	Teaching Assistant The University of Western Ontario 2021 - 2022