Fairness-Oriented Entity Resolution Tool for Streaming Data

Tiago Brasileiro Araújo[1,2] **, Vasilis Efthymiou**[3,4] **, Kostas Stefanidis**[2] **,**
Rafael de Souza Guerra[5]

[1]Federal Institute of Paraíba, Soledade, Brazil

tiago.brasileiro@ifpb.edu.br

[2]Tampere University, Tampere, Finland

{tiago.brasileiroaraujo, konstantinos.stefanidis}@tuni.fi

[3]Harokopio University of Athens, Athens, Greece

vefthym@hua.gr

[4]FORTH-ICS, Heraklion, Greece

[5]Federal University of Campina Grande, Campina Grande, Brazil

rafael.guerra@ccc.ufcg.edu.br

Abstract. *Entity Resolution (ER) plays a crucial role, facilitating the integration of knowledge bases and identifying similarities among entities from different sources. In this work, we address the following challenges: streaming data, incremental processing, and fairness. There is a lack of studies involving fairness and ER, which is related to the absence of discrimination or bias. Considering this context, this work presents TREATS*[1]*, a fairness-aware ER tool able to deal with streaming and incremental data, which goes beyond matching based on the similarity scores and also applies to target fairness constraints. Overall, our contributions aim to advance the field of ER by offering a matching tool that considers both technical challenges and ethical considerations.*

1. Introduction and Motivation

The extensive volume of records gathered from various origins mandates an initial phase of consolidation and cleansing before initiating any data analysis pursuits [Christophides et al. 2015]. In this context, Entity Resolution (ER) emerges as a crucial task, facilitating the integration of diverse knowledge bases or identifying similarities among entities. The task of ER has its own characteristics, challenges, and features. In this work, we address the following challenges: streaming data, incremental processing, and fairness.

Streaming data pertains to data originating from dynamic sources like web systems, social media platforms, and sensors, continuously updated [Araújo et al. 2022]. Real-world scenarios such as news, social media, e-commerce, and airline security exemplify streaming scenarios due to their dynamic nature. Incremental processing, particularly in ER, involves the reception of data in a continuous stream and selective reprocessing of the ER task. This process must account for previously compared entities and

[1]**Link to the presentation:** https://youtu.be/X5vRyEu8FQg

newly arrived entities in each increment. Fairness is fundamentally linked to the absence of discrimination or bias. Such bias may arise from the algorithm itself, potentially reflecting the commercial or personal preferences of its developers [Shahbazi et al. 2023]. Therefore, fairness criteria aim to minimize the consequences of data bias in ER systems, demanding more than just accuracy optimization, as traditionally practiced in ER. For example, in the context of airline security, matching records related to a no-fly list and the passenger list should acknowledge that these two lists may have different demographic distributions, thus necessitating the avoidance of bias against specific groups of passengers. Moreover, it can originate from the data itself, for instance, if a survey incorporates biased questions or if a specific demographic is misrepresented in the input data [Pitoura et al. 2022].

ER pipelines generally consist of a blocking step and a matching step [Christophides et al. 2021]. The blocking step aims to reduce the computational cost of comparing all possible pairs of entities by grouping similar entities into blocks, thus limiting comparisons within each block and significantly enhancing efficiency. The matching step then involves comparing the entities within each block using various functions to assess their similarity. This work focuses on the matching step, handling candidate pairs from the blocking step in a streaming manner, determining their match likelihood through similarity scores, and incrementally returning the most promising candidates while also considering fairness aspects.

Recognizing these challenges, we have developed a novel tool called TREATS – available at https://github.com/brasileiroaraujo/FAIR – which involves *sTREaming* data and *fAirness* criteria in the *enTity reSolution* context. TREATS is a fairness-aware ER tool able to deal with streaming and incremental data, which goes beyond matching based on the similarity scores and also applies to target fairness constraints among multiple groups of interest. For instance, let gender be considered as the sensitive attribute; the groups of interest may be determined by: *male*, *female*, *transgender*, and *others*[2]. Therefore, the matching output should not only consider the similarity scores of entities but also satisfy equal representation among the groups.

2. System Design and Implementation

When handling streaming data, it is essential to incorporate new components before the steps in the traditional ER workflow. These components are tasked with organizing the micro-batches that are slated for processing. To this end, the component known as sender handles candidate (i.e., pairs of entities) groups, forwarding them to the comparison step. Since streaming data are being constantly received, the ER steps should be performed according to a time budget. This dynamic poses a challenge for the workflow, necessitating swift execution. To address this challenge, parallel processing strategies were applied to enhance ER efficiency without having a negative impact on effectiveness. More specifically, in this work, we applied Graphics Processing Unit (GPU) environments during the comparison and classification step to guarantee efficiency and also applied high-quality ER frameworks, such as Ditto [Li et al. 2020] and GNEM [Chen et al. 2021]. In this sense, GPU stands for parallel processing environment of general purpose able to handle multiple tasks or calculations simultaneously. In Figure 1, the GPU environment is

[2]Note, entities that do not present any of these values are inserted into the group classified as *others*.

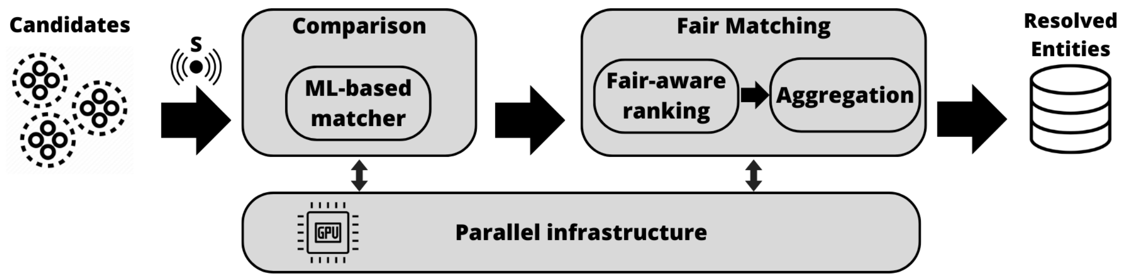


Figure 1. Workflow considering streaming data and fairness aspects.

illustrated by the parallel infrastructure component.

Comparison step. A Machine Learning (ML) matcher component is applied to compare the candidates (i.e., entity pairs) and predict the matching score between them. Considering ML approaches, Ditto and GNEM achieved high-quality results in different benchmark data sources with significantly less training data. These results can be attributed to the improved language understanding capability mainly through pre-trained language models. Note that different ML matchers can be applied to TREATS. To perform the comparison step, and consequently execute the matcher, a GPU infrastructure is used. With the pre-trained models, the sender component consumes the data provided by the data sources in a streaming way, buffers it in micro-batches, and sends it to the matcher. To connect the sender to the matcher, a streaming processing platform, namely Apache Kafka, is applied.

Fair matching. For each micro-batch (i.e., increment), the candidates, whose matching scores were computed in the comparison step, are ranked and resolved in the fair-matching step. The fair-matching step is divided into two substeps: i) Fairness-aware ranking, where the ranking method is applied to the candidates, taking into account the similarity scores and the fairness criteria to define the matches; ii) Aggregation, which is responsible for aggregating the current matches with the matches resolved in prior increments. Following, we describe how the fair-matching step works in the sense of ranking and resolving candidates in the context of streaming data and fairness constraints. In addition to performing the ranking, fair matching is also responsible for resolving the entities. [Shahbazi et al. 2023] highlight potential unfairness in real-world societies and data under two common conditions: (i) over-representation of certain demographic groups, and (ii) higher name similarity within some groups compared to others. For this reason, the matches are defined based on the rank position, maximizing the cumulative scores and satisfying the fairness constraints.

In this study, we expand the conventional assessment of ER results to include fairness considerations. In a fair ER context, the initial comparison results should not only be the most likely matches but also adhere to specified fairness criteria. We extend the ranking method introduced by [Efthymiou et al. 2021] to handle streaming data incrementally and consider multiple groups of interest, beyond the original method's focus on a single attribute with binary values (i.e., only consider two groups: protected and non-protected). Following the TREATS workflow, the proposed fairness-aware ranking algorithm takes as input the output from the comparison step, which consists of a list of candidates with their respective similarity scores. Initially, the fairness-aware ranking algorithm sorts this list in descending order based on the scores, ensuring that candidates with higher similarity scores are prioritized. Consequently, the set of candidates is converted into priority queues sorted by descending scores, with each iteration of the queue returning the top pair as a match. Since the ranking method handles multiple groups of interest, the *id* (i.e., key) of each group is extracted from the attribute values of the entity pairs that make up the candidate. Thus, each candidate is classified into a group of interest based on the value of the sensitive attribute. Then, the set of candidates is iterated, extracting the group *id* for each candidate and inserting the candidate into the appropriate priority queue based on its group *id*.

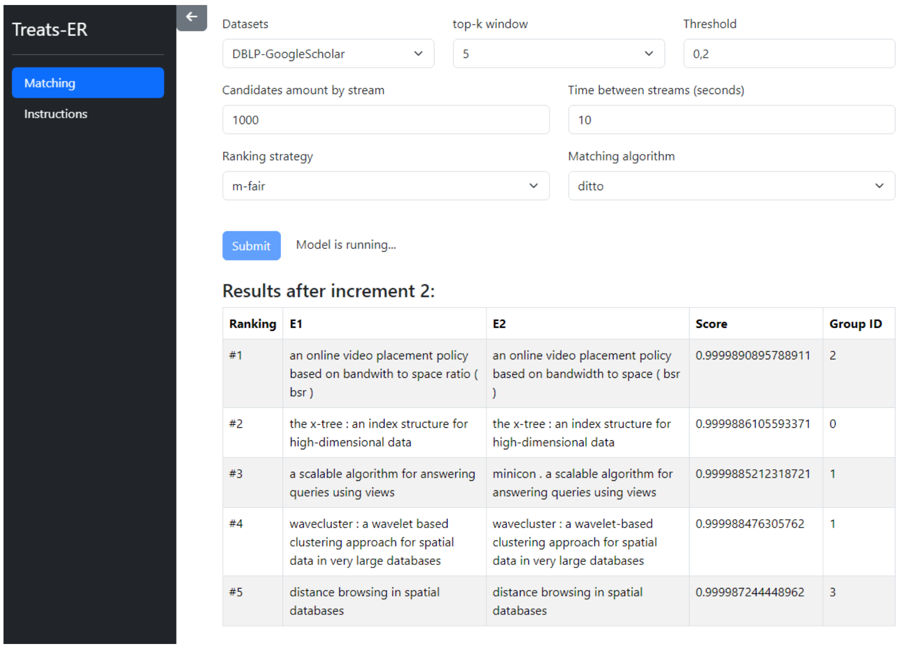
In a streaming setting, the number or size of input increments is unknown in advance. Therefore, we propose that a ranking-based matching approach, which focuses on finding a subset of potentially infinite matches, with a fixed size k, is the most realistic approach for matching. Hence, we consider the matching task as a top-k ranking problem. Following the fairness-aware ranking, while the queues of candidates are not empty and the number of matches is less than k, the algorithm picks the top candidate from a different priority queue each time, following the sequence of the groups. The fairness constraint is used to guide the algorithm, ensuring it avoids unfair or discriminatory disparities among the groups. Thus, the fair ranking process aims to select one candidate from each group per iteration.

TREATS web interface.[3] Figure 2 presents the web interface developed for TREATS. The black bar on the left side of the interface is the navigation menu. This navigation bar allows users to easily switch between the matching functionality (i.e., execute TREATS) and obtain guidance on using the interface. Regarding the parameters, there are seven: i) Datasets: allow the user to select a pair of datasets from pre-loaded options; ii) Top-k Window: specifies the number of top results to display after each increment; iii) Threshold: sets the similarity score threshold for considering matches during the comparison step; iv) Candidates Amount by Stream: determines the number of candidate pairs to be compared in each streaming increment; v) Time Between Streams (in seconds): defines the interval time in seconds between each data stream increment; vi) Ranking Strategy: Offers the user the option to select the approach for ranking the results; vii) Matching Algorithm: indicates the algorithm used for matching entity pairs (e.g., Ditto or GNEM). After defining the parameters, the "Submit" button initiates the TREATS workflow. The results are displayed in a tabular format with the following columns: Ranking (the position in the queue), E_1 (entity information from dataset 1), E_2 (entity information from dataset 2), Score (similarity score between the entities), and Group ID (id that represents the group of interest). For each increment, the table is updated to include the data from the current increment while also considering the data from previous increments.

3. Demonstration Highlights

TREATS is running on a Cloud GPU instance in the Google Cloud platform, which offers robust scalability and reliability for handling large volumes of streaming data, with the following configuration: N1-standard-4 instance with 2 NVIDIA T4 GPUs, 4 vCPUs (2 cores each), and 15 GB memory. All the steps of the proposed tool were implemented

[3]The TREATS interface can also be viewed in operation through the shared video and it can be reproduced from the publicly available code on GitHub.



Datasets: DBLP-GoogleScholar
top-k window: 5
Threshold: 0,2
Candidates amount by stream: 1000
Time between streams (seconds): 10
Ranking strategy: m-fair
Matching algorithm: ditto

Model is running...

Results after increment 2:

Ranking	E1	E2	Score	Group ID
#1	an online video placement policy based on bandwith to space ratio (bsr)	an online video placement policy based on bandwidth to space (bsr)	0.9999890895788911	2
#2	the x-tree : an index structure for high-dimensional data	the x-tree : an index structure for high-dimensional data	0.9999886105593371	0
#3	a scalable algorithm for answering queries using views	minicon . a scalable algorithm for answering queries using views	0.9999885212318721	1
#4	wavecluster : a wavelet based clustering approach for spatial data in very large databases	wavecluster : a wavelet-based clustering approach for spatial data in very large databases	0.999988476305762	1
#5	distance browsing in spatial databases	distance browsing in spatial databases	0.999987244448962	3

Figure 2. Web interface of TREATS.

in Python. To evaluate the TREATS tool regarding effectiveness and fairness, we used five real-world pairs publicly available: DBLP-GoogleScholar, DBLP-ACM, Amazon-Google, Walmart-Amazon, and iTunes-Amazon. The TREATS interface was developed using React[4], a popular JavaScript library for building user interfaces. React is renowned for its efficiency and flexibility, allowing developers to create large web applications that can update and render efficiently in response to data changes. Furthermore, the interface, combined with the cloud environment, provides TREATS with the advantages of cloud scalability and performance, delivering a smooth and responsive user experience. This setup makes it easier to manage the dynamic and incremental nature of streaming data in ER tasks. Following, the main highlights of TREATS in terms of effectiveness, efficiency, and fairness.

Effectiveness. The metric employed is $Precision$, which measures the correctness of the returned matches with respect to the ground truth of known matches. Considering the data sources Amazon-Google and Walmart-Amazon, when Ditto is applied, TREATS achieved a precision of 0.95 and 0.85 respectively while it achieved a precision of 1 for the other data sources. When GNEM is applied, TREATS achieved a precision of 0.85 and 0.95 for the data sources Amazon-Google and Walmart-Amazon respectively[5].

Efficiency. To assess the efficiency of TREATS using Ditto and GNEM, we measured the execution time. Across all scenarios investigated, TREATS with Ditto exhibited execution times ranging from approximately 2 to 5 seconds, while TREATS with GNEM had execution times ranging from approximately 4 to 12 seconds.

Fairness. To evaluate the fairness of TREATS with the application of Ditto and GNEM over the five data source pairs, we apply the Bias metric, which evaluates dispar-

[4]https://react.dev/
[5]GNEM provides pre-trained models only for the Amazon-Google and Walmart-Amazon data sources.

ities in predictions across two groups in the results of an ER output (seen as a ranking problem). A value of Bias equal to zero implies no bias (i.e., all groups are represented equally in the results) and higher values indicate stronger bias in favor of or against one of the groups. TREATS achieved the perfect score of Bias = 0 for Amazon-Google and Walmart-Amazon data sources when Ditto is applied. When GNEM is applied in TREATS, it achieves the perfect score of Bias = 0 for Amazon-Google and Bias = 0.09 for Walmart-Amazon. This behavior suggests that the application of the fairness-aware ranking method improves the result in terms of mitigating bias. Regarding the other data sources, TREATS presents Bias around 0.25 for DBLP-GoogleScholar and Bias = 0 for DBLP-ACM and iTunes-Amazon.

4. Conclusion

The proposed fairness-aware ER tool setting addresses constraints among multiple groups of interest, offering a robust and equitable solution to ER challenges. The experimental evaluation provides insights into the impact of fairness-aware ranking methods on ER systems, contributing to the ongoing discourse on fairness in ER. Overall, our contributions aim to advance the field of ER by offering a workflow that considers both technical challenges (i.e., streaming data and incremental processing) and ethical considerations (i.e., fairness).

Acknowledgement. This work has received funding from the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Technology (GSRT), under grant agreement No 969.

References

Araújo, T. B., Stefanidis, K., Pires, C. E. S., Nummenmaa, J., and da Nóbrega, T. P. (2022). Incremental entity blocking over heterogeneous streaming data. *Information*, 13(12):568.

Chen, R., Shen, Y., and Zhang, D. (2021). Gnem: a generic one-to-set neural entity matching framework. In *Proceedings of the Web Conference 2021*, pages 1686–1694.

Christophides, V., Efthymiou, V., Palpanas, T., Papadakis, G., and Stefanidis, K. (2021). An overview of end-to-end entity resolution for big data. *ACM Comput. Surv.*, 53(6):127:1–127:42.

Christophides, V., Efthymiou, V., and Stefanidis, K. (2015). Entity resolution in the web of data. *Synthesis Lectures on the Semantic Web*, 5(3):1–122.

Efthymiou, V., Stefanidis, K., Pitoura, E., and Christophides, V. (2021). FairER: entity resolution with fairness constraints. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3004–3008.

Li, Y., Li, J., Suhara, Y., Doan, A., and Tan, W.-C. (2020). Deep entity matching with pre-trained language models. *Proceedings of the VLDB Endowment*, 14(1):50–60.

Pitoura, E., Stefanidis, K., and Koutrika, G. (2022). Fairness in rankings and recommendations: an overview. *The VLDB Journal*, pages 1–28.

Shahbazi, N., Danevski, N., Nargesian, F., Asudeh, A., and Srivastava, D. (2023). Through the fairness lens: Experimental analysis and evaluation of entity matching. *Proceedings of the VLDB Endowment*, 16(11):3279–3292.