BipartiteJoin: Optimal Similarity Join for Fuzzy Bipartite Matching

Ondrej Rozinek[1], Monika Borkovcova[2], and Jan Mares[1,3]

[1] University of Pardubice, Department of Process Control, Studentska 95, 532 10 Pardubice, Czech Republic,
ondrej.rozinek@gmail.com,
[2] University of Pardubice, Department of Information Technology, Studentska 95, 532 10 Pardubice, Czech Republic,
[3] University of Chemistry and Technology Prague, Department of Mathematics, Informatics and Cybernetics, Technicka 5, 166 28 Prague, Czech Republic

Abstract. Set similarity join, crucial for data cleaning, integration, and recommendation systems, identifies set pairs exceeding a similarity threshold. Our approach combines a count Q-gram filter with maximum weighted bipartite matching, balancing accuracy and efficiency. The Q-gram filter, based on the relationship between Q-gram similarity and edit distance, reduces the number of comparisons, operating in constant time on a pre-built index. This enables real-time processing, as only a minimal number of pairs are verified through Fuzzy Bipartite Matching, significantly enhancing the efficiency of similarity joins.

Keywords: similarity join, Q-gram filter, record linkage, entity resolution, similarity space, bipartite matching

1 Introduction

Set similarity join identifies all pairs of sets within a single record collection or across two different collections when the similarity score is above a certain threshold α [3, 5, 7]. This process is an essential operation in many applications, such as data cleaning and integration, personalized recommendation and record deduplication.

As shown in the Fig. 2, our focus is on a two-step method for similarity join: firstly, using a count Q-gram filter, and secondly, employing maximum weighted bipartite matching as the best approach for solving the combinatorial assignment problem [10]. This method involves fuzzy token similarity in bipartite matchings, recognized state-of-the-arts for its high accuracy in classifying records as matches or non-matches in an error-tolerant manner. However, this approach results in polynomial time complexity, $\mathcal{O}(n^3)$, which is managed by the Kuhn-Munkres algorithm (Fig. 1).

To improve the efficiency of similarity joins, we avoid exhaustive pair comparisons by applying a count Q-gram filter. In real-world applications, Ukkonen's lemma [9, 12] is used to establish a direct relation between Q-gram similarity and

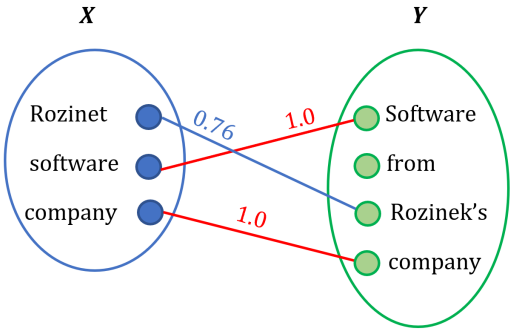

Fig. 1. Maximum weighted bipartite matching of two records $\mathcal{X}$ and $\mathcal{Y}$.

edit distance (Levenshtein), which acts as a rigorous mathematical filter to reduce the number of comparisons without missing any true matches. As indicated in the figure, the Q-gram filter operates in constant time, $\mathcal{O}(1)$ on a pre-built inverted Q-gram index. This significant reduction in comparison pairs enables real-time processing, as only a small fraction of candidate pairs is further verified for actual matches in the second stage of Fuzzy Bipartite Matching (Fig. 2).

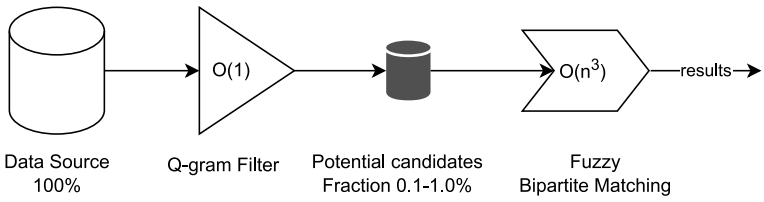

Fig. 2. Block diagram of the processing of records from the source in real-time by a two-step system of Q-gram filter and Fuzzy Bipartite Matching

According to Ukkonen's lemma [9, 12], let X and Y be tokens with the edit distance $d(X, Y)$. Then, the Q-gram similarity $|Q_X \cap Q_Y|$ of the tokens X and Y is at least $t = \inf_d\{|Q_X \cap Q_Y|\} = \max\{|X|, |Y|\} - q + 1 - qd(X, Y)$, where t is a Q-gram similarity threshold with respect to $d(X, Y)$ and q is the Q-gram length.

The problem often arises with the assumption that this constraint is applied to the entire record [12]. Consider two records $\mathcal{X}$ and $\mathcal{Y}$, split into sets of tokens $X_i \in \mathcal{X}$ and $Y_j \in \mathcal{Y}$. This simplification is demonstrated in Example 1, leading to differing results of the Q-gram filter as a constraint of fuzzy bipartite matching. We address this gap in our article's goal.

Example 1. Consider two records $\mathcal{X}$ and $\mathcal{Y}$, and their token sets, $\{X_1, X_2\}$ and $\{Y_1, Y_2\}$. Assume a matching set of token pairs $\mathcal{M}$, such that $M = \{\{X_1, Y_1\}, \{X_2, Y_2\}\}$.

Then, for the entire record, we have $t = \max\{|X_1 \cup X_2|, |Y_1 \cup Y_2|\} - q + 1 - qd(X_1 \cup X_2, Y_1 \cup Y_2)$, which is different from the calculation on the matching pairs separately $t_{\mathcal{M}} = \max\{|X_1|, |Y_1|\} - q + 1 - qd(X_1, Y_1) + \max\{|X_2|, |Y_2|\} - q + 1 - qd(X_2, Y_2)$, given by edges in maximum weighted bipartite matching. Hence, we prove that $t \neq t_{\mathcal{M}}$ and a more precise filter for $t_{\mathcal{M}}$ must exist.

2 Related Work

Recent research surveys, such as the one by Papadakis et al. (2020) [7], have identified state-of-the-art methods for set similarity joins based on edit constraints, including FastJoin, SilkMoth, and MF-Join.

FastJoin [10], the initial method in this domain, is based on the principle that two sets with a bipartite matching score of at least α must have at least $\lceil \alpha \rceil$ shared tokens. For a set $\mathcal{X}$ with n tokens, any $n - \lceil \alpha \rceil + 1$ tokens form its signature. If another set $\mathcal{Y}$ lacks these signature tokens, a match with $\mathcal{X}$ is improbable. FastJoin essentially adapts prefix filtering to fuzzy criteria.

SilkMoth [2] enhances FastJoin's signature scheme, reducing token count to limit candidate sets. It applies refinement filters like Check Filter (CF) and Nearest Neighbor Filter (NNF) for sets $\mathcal{Y}$ containing a signature token. CF calculates a similarity threshold for each element in $\mathcal{X}$, pruning $\mathcal{Y}$ if no element pair meets this threshold. NNF pairs each $\mathcal{X}$ element with its closest in $\mathcal{Y}$, setting an upper limit for matching scores.

MF-Join [11] diverges by using two thresholds: δ_1 for sets and δ_2 for elements. It considers element pairs above δ_2 for matching, but manually setting these thresholds is challenging.

TokenJoin [13] proposes a lightweight and effective token-based filtering approach that accelerates the speed of similarity joins.

Unlike other methods, our approach aims to use a count Q-gram filter [7] to make the similarity join process quicker in maximum weighted bipartite matching. Our main goal is to have a precise filter that is reliable in the pruning process, rather than a faster filter that is less accurate.

3 Optimal Count Q-gram Filter

Consider the edit distance between two strings X and Y, denoted as $d(X, Y)$, and the worst case of the expected distance function represented as $d(\alpha, |X|, |Y|)$. If we use the floor function, denoted by $\lfloor . \rfloor$, we find that the normalized similarity metric $s_n(X, Y)$, with $\alpha \in [0, 1] \subset \mathbb{R}$ as a threshold, satisfies the following relationship:

$$s_n(X, Y) \geq \alpha \iff d(X, Y) \leq d(\alpha, |X|, |Y|) = \left\lfloor \frac{1 - \alpha}{1 + \alpha}(|X| + |Y|) \right\rfloor. \quad (1)$$

This relationship is derived from the general understanding of distance and similarity metrics [8]. When substituting self-similarities, which equal the corre-

sponding cardinality of the sets, as $s(X, X) = |X|$ and $s(Y, Y) = |Y|$, the expected distance $d(\alpha, |X|, |Y|)$ is maximized when the similarity is at its minimum, bound by the threshold α. This condition is met if and only if $s_n(X, Y) = \alpha$:

$$\begin{aligned} d(X,Y) = \lfloor d_R \rfloor &= \left\lfloor \frac{1 - s_n(X,Y)}{1 + s_n(X,Y)} (s(X,X) + s(Y,Y)) \right\rfloor \\ &\leq \sup_{\alpha} d(X,Y) = \left\lfloor \frac{1-\alpha}{1+\alpha} (|X| + |Y|) \right\rfloor = d(\alpha, |X|, |Y|). \end{aligned} \tag{2}$$

Given that the edit distance is an integer, the floor function is used to discard the fractional part, ensuring the measure remains defined and accurate.

In our refinement of the count Q-Gram filter model, we introduce more precise assumptions about the token sets $\mathcal{X}$ and $\mathcal{Y}$, as detailed in the Example 1 provided. This leads to the derivation of an optimized filter, specifically designed to retain every comparison pair where Fuzzy Bipartite Matching exceeds the threshold α.

Central to our approach is the assumption that the matching token pairs from set $\mathcal{M}$ are known to us, but their exact edit distances remain undetermined. Despite this, we can approximate these edit distances by relying on predictions based on expected edit distances. This predictive approach allows us to effectively gauge similarity without the need for precise edit distance values, aligning with our refined model's objectives.

The optimality of the filter is understood from the perspective that it represents the infimum of Q-gram similarity, and consequently, it is the optimal edit constraint for maximum weighted bipartite matching.

Theorem 1 (Optimal Count Q-gram Filter for Bipartite Matching). *Let $\mathcal{X}$ and $\mathcal{Y}$ be records representing a set of tokens. Then the Q-gram similarity in bipartite matching of $\mathcal{X}$, $\mathcal{Y}$ and cardinality $|\mathcal{M}|$ for a given threshold $s_n(\mathcal{X}, \mathcal{Y}) \geq \alpha$ is at least*

$$\begin{aligned} t_{\mathcal{M}} &= \inf_{\alpha} \{|Q_{\mathcal{X}} \cap Q_{\mathcal{Y}}|\} \\ &= \underbrace{\sum_{(i,j)\in\mathcal{M}} \max\{|X_i|, |Y_j|\} - |\mathcal{M}|q + |\mathcal{M}|}_{\textit{maximum shared Q-grams}} \underbrace{- q \max_{\alpha} \sum_{(i,j)\in\mathcal{M}} d(\alpha_{i,j}, |X_i|, |Y_j|)}_{\textit{loss function}}, \end{aligned} \tag{3}$$

containing a linear combination of

$$d(\alpha_{i,j}, |X_i|, |Y_j|) = \frac{1 - \alpha_{i,j}}{1 + \alpha_{i,j}} (|X_i| + |Y_j|) \tag{4}$$

under the constraint α for which the linear combination is maximized.

Proof. Consider the sum over connected pairs of tokens with cardinality $|\mathcal{M}|$

$$\begin{aligned}
&\inf_{\alpha}\{|Q_{\mathcal{X}} \cap Q_{\mathcal{Y}}|\} = \inf_{\alpha}\left\{\sum_{(i,j)\in\mathcal{M}} |Q_{X_i} \cap Q_{Y_i}|\right\} = \sum_{(i,j)\in\mathcal{M}} \inf_{\alpha_{i,j}}\{|Q_{X_i} \cap Q_{Y_j}|\} \\
&= \sum_{(i,j)\in\mathcal{M}} \inf_{\alpha_{i,j}}\{\max\{|X_i|,|Y_j|\} - q + 1 - qd(X,Y)\} \\
&= \sum_{(i,j)\in\mathcal{M}} \{\max\{|X_i|,|Y_j|\} - q + 1 - q\sup_{\alpha_{i,j}} d(X,Y)\} \\
&= \sum_{(i,j)\in\mathcal{M}} \max\{|X_i|,|Y_j|\} - |\mathcal{M}|q + |\mathcal{M}| - q\max_{\alpha} \sum_{(i,j)\in\mathcal{M}} d(\alpha_{i,j},|X_i|,|Y_j|).
\end{aligned} \tag{5}$$

Each $\alpha_{i,j}$ is the distributed minimum similarity for each token, giving a threshold vector that should maximize the sum of the expected distances $d(\alpha_{i,j},|X_i|,|Y_j|)$ so that $s_n(\mathcal{X},\mathcal{Y}) \geq \alpha$ holds for t_M. Formalizing this, we get the task

$$\begin{aligned}
&\text{maximize} \sum_{(i,j)\in\mathcal{M}} d(\alpha_{i,j},|X_i|,|Y_j|), \\
&\text{subject to} \sum_{(i,j)\in M} \alpha_{i,j} \geq \text{(Table 1)} \qquad \alpha \in [0,1],\, i = 1,\ldots,|\mathcal{M}|, \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \alpha_{i,j} \in [0,1],\, j = 1,\ldots,|\mathcal{M}|.
\end{aligned}$$

This leads to an integer linear programming task equivalent to the Knapsack problem, solvable in $\mathcal{O}(nb)$ time. The optimization algorithm determines the maximum expected edit distance distribution across tokens, maintaining the similarity threshold $s_n(\mathcal{X},\mathcal{Y}) \geq \alpha$.

Table 1. Overview of fuzzy token similarity functions and their corresponding constraints [6] on the integer linear programming problem.

Similarity Measure	**Subject to** $\sum_{(i,j)\in M} \alpha_{i,j} \geq$
Fuzzy Dice	$\frac{\alpha}{2}(\lvert\mathcal{X}\rvert + \lvert\mathcal{Y}\rvert)$
Fuzzy Cosine	$\alpha\sqrt{\lvert\mathcal{X}\rvert\lvert\mathcal{Y}\rvert}$
Fuzzy Jaccard	$\frac{\alpha}{1+\alpha}(\lvert\mathcal{X}\rvert + \lvert\mathcal{Y}\rvert)$
Fuzzy Overlap	$\alpha\min\{\lvert\mathcal{X}\rvert,\lvert\mathcal{Y}\rvert\}$

4 Approximate Count Q-gram Filter

Let F_x and F_Y be discrete distribution functions of ascending sorted lengths $|X_i|$ and $|Y_i|$. Then the Q-gram similarity in bipartite matching for unknown

connected edges of records $\mathcal{X}$, $\mathcal{Y}$ and cardinality $|\mathcal{M}|$ is at least

$$\begin{aligned} t_{\mathcal{M}} \approx \widehat{t}_{\mathcal{M}} = & \frac{2q\alpha + \alpha - 2q + 1}{2 + \alpha}(F_X[|\mathcal{M}|] + F_Y[|\mathcal{M}|]) \\ & + \frac{1}{2}\Big|F_X[|\mathcal{M}|] - F_Y[|\mathcal{M}|]\Big| - |\mathcal{M}|q + |\mathcal{M}| \end{aligned} \tag{6}$$

for a classification Fuzzy Bipartite Matching threshold $s_n(\mathcal{X}, \mathcal{Y}) \geq \alpha$.

The derivation results from a new approximation method that involves using several techniques to establish a less tight lower bound for certain terms. To aid the reader's understanding, we first present the following equations that are integral to the final deduced formula.

$$\max\left\{\sum_{(i,j)\in\mathcal{M}} |X_i|, \sum_{(i,j)\in\mathcal{M}} |Y_j|\right\} \leq \sum_{(i,j)\in\mathcal{M}} \max\{|X_i|, |Y_j|\}. \tag{7}$$

We can also express the maximum of any two variables $a, b \in \mathbb{R}$ in another analytical form:

$$\max\{a, b\} = \frac{1}{2}(a + b + |a - b|), \tag{8}$$

and now define the cumulative sum (discrete distribution function) of ascending sorted length F_X and F_Y. Finally, we obtain the inequality

$$\max\{F_{\mathcal{X}}[|\mathcal{M}|], F_{\mathcal{Y}}[|\mathcal{M}|]\} \leq \max\left\{\sum_{(i,j)\in\mathcal{M}} |X_i|, \sum_{(i,j)\in\mathcal{M}} |Y_j|\right\}. \tag{9}$$

With equations (7), (8), and (9), we proceed to the full derivation, assuming the constancy of $\alpha_{i,j} = \alpha$ for simplicity. We also apply the floor function $\lfloor . \rfloor$ to maintain integer values for shared Q-grams. The entire derivation is as follows:

$$
\begin{aligned}
t_{\mathcal{M}} &= \sum_{(i,j)\in\mathcal{M}} \max\{|X_i|,|Y_j|\} - |\mathcal{M}|q + |\mathcal{M}| - q \max_{\alpha} \sum_{(i,j)\in\mathcal{M}} d(\alpha,|X_i|,|Y_j|) \\
&= \sum_{(i,j)\in\mathcal{M}} \max\{|X_i|,|Y_j|\} - |\mathcal{M}|q + |\mathcal{M}| - q \max_{\alpha} \sum_{(i,j)\in\mathcal{M}} \frac{1-\alpha_{i,j}}{1+\alpha_{i,j}}(|X_i|+|Y_j|) \\
&\approx \sum_{(i,j)\in\mathcal{M}} \max\{|X_i|,|Y_j|\} - |\mathcal{M}|q + |\mathcal{M}| - q\frac{1-\alpha}{1+\alpha} \sum_{(i,j)\in\mathcal{M}} (|X_i|+|Y_j|) \\
&\geq \max\left\{ \sum_{(i,j)\in\mathcal{M}} |X_i|, \sum_{(i,j)\in\mathcal{M}} |Y_j| \right\} - |\mathcal{M}|q + |\mathcal{M}| - \frac{q-q\alpha}{1+\alpha} \sum_{(i,j)\in\mathcal{M}} (|X_i|+|Y_j|) \\
&= \frac{1}{2} \sum_{(i,j)\in\mathcal{M}} (|X_i|+|Y_j|) + \frac{1}{2} \left| \sum_{(i,j)\in\mathcal{M}} |X_i| - \sum_{(i,j)\in\mathcal{M}} |Y_k| \right| - |\mathcal{M}|q + |\mathcal{M}| - \frac{q-q\alpha}{1+\alpha} \sum_{(i,j)\in\mathcal{M}} (|X_i|+|Y_j|) \\
&= \frac{2q\alpha+\alpha-2q+1}{2+\alpha} \sum_{(i,j)\in\mathcal{M}} (|X_i|+|Y_j|) + \frac{1}{2} \left| \sum_{(i,j)\in\mathcal{M}} |X_i| - \sum_{(i,j)\in\mathcal{M}} |Y_k| \right| - |\mathcal{M}|q + |\mathcal{M}| \\
&\geq \frac{2q\alpha+\alpha-2q+1}{2+\alpha} (F_X[|\mathcal{M}|] + F_Y[|\mathcal{M}|]) + \frac{1}{2} \left| F_X[|\mathcal{M}|] - F_Y[|\mathcal{M}|] \right| - |\mathcal{M}|q + |\mathcal{M}| \\
&\geq \left\lfloor \frac{2q\alpha+\alpha-2q+1}{2+\alpha} (F_X[|\mathcal{M}|] + F_Y[|\mathcal{M}|]) + \frac{1}{2} \left| F_X[|\mathcal{M}|] - F_Y[|\mathcal{M}|] \right| \right\rfloor - |\mathcal{M}|q + |\mathcal{M}| \\
&= \widehat{t}_{\mathcal{M}} \\
&\implies t_{\mathcal{M}} \approx \widehat{t}_{\mathcal{M}}.
\end{aligned}
\tag{10}
$$

Consequently, when utilizing a pre-built inverted Q-gram index that includes the distribution of ascending sorted lengths, we can achieve a time complexity of $\mathcal{O}(1)$.

5 Experiments

We computed the *non-interpolated average precision* of this ranking. According to the papers [1, 4], we calculate the precision and recall as:

$$\text{Precision} = \frac{c(i)}{i}, \tag{11}$$

$$\text{Recall} = \frac{c(i)}{m}, \tag{12}$$

where $c(i)$ is the number of correct matching pairs ranked before position i, and m is total number of correct matches. Consequently *interpolated precision* at recall r is the $\max_i \frac{c(i)}{i}$, where the max is taken over all ranks i such that $\frac{c(i)}{m} \geq r$. The overall relative performance of the compared similarity functions is calculated using the maximum F1-score as:

$$\text{F1-score} = 2 \times \frac{(\text{Precision} \times \text{Recall})}{\text{Precision+Recall}}, \tag{13}$$

and shown in the Tab. 3. The table shows that the best results, with an accuracy of 85.09%, were achieved using the Fuzzy Overlap method based on maximum weighted bipartite matching on its own, and slightly lower at 85.01% when

combined with the Q-gram filter. These results support the idea that the approximated optimal Q-gram filter is highly accurate and confirm that our way of approximating it mathematically is correct. However, it's worth mentioning that even though this is an approximation, a few records were found in a detailed analysis that didn't pass through the filter. Still, the difference in the F-score is very small, only 0.08%.

The Q-gram filter and Fuzzy Overlap, when used together, enable real-time fuzzy matching. This combination operates on a pre-built inverted Q-gram index and completes the task in just 220 ms, which is a speed increase by an order of magnitude compared to running Fuzzy Overlap alone, which takes 13s:426ms. Note that the speed improvement is influenced by the α threshold parameter, affecting the size of the potential candidate pool in the second stage of Fuzzy Overlap (refer to Figure 2). Testing to evaluate the relative time complexity was performed on a single-core Intel i7 11370H processor with a maximum turbo frequency of 4.80 GHz and 16GB of RAM.

Name	**Number of strings**	**Name**	**Number of strings**
Animal	5,709	Game	911
Bird Kunkel	336	Park	654
Bird Nybird	982	Restaurant	863
Bird Scott1	38	Ucd-people	90
Bird Scott2	719	Census	841
Business	2,139		

Table 2. Datasets used in experiments from original sources [1]

Similarity	**F1-score**	**Similarity**	**F1-score**
Fuzzy Overlap	85.09 %	Smith-Waterman	75.71 %
approx. 3-gram filter+Fuzzy Overlap	85.01 %	Smith-Waterman-Gotoh	75.54 %
approx. 2-gram filter+Fuzzy Overlap	84.88 %	Jaro	75.29 %
Fuzzy Jaccard (Levenshtein $\delta = 0.8$))	84.17 %	Overlap 3-gram	73.21 %
Jaro-Winkler	81.45 %	Jaccard 2-gram	71.05 %
L2 Monge-Elkan (Levenshtein)	80.80 %	Dice 2-gram	71.05 %
Damerau-Levenshtein	76.86 %	Jaccard 3-gram	70.86 %
Levenshtein	76.83 %	Dice 3-gram	70.86 %
Needleman-Wunsch	76.25 %	Overlap 2-gram	66.92 %

Table 3. Comparison of selected similarity functions ranked in descending order of F1-score

Similarity	**Elapsed Time**	**Similarity**	**Elapsed Time**
Levenshtein	13s:426ms	L2 Monge-Elkan (Levenshtein)	14s:209ms
Damerau-Levenshtein	22s:824ms	Jaccard 2-gram	10s:542ms
Jaro	3s:902ms	Jaccard 3-gram	9s:829ms
Jaro-Winkler	3s:772ms	Dice 2-gram	11s:95ms
Needleman-Wunsch	28s:170ms	Dice 3-gram	10s:717ms
Smith-Waterman	28s:600ms	Overlap 3-gram	10s:251ms
Fuzzy Overlap	13s:474ms	Overlap 2-gram	11s:549ms
Q-Gram Filter+Fuzzy Overlap	0s:220ms	Fuzzy Jaccard ($\delta = 0.8$)	12s:824ms

Table 4. Relative Time Complexity

6 Conclusion

In this research, we conducted a detailed analysis of different similarity functions. We compared these functions using metrics such as precision, recall, and F1-score. Specifically, our focus was on evaluating our similarity join method, which is based on a two-step approach. This approach combines an approximated Q-gram count filter with the Fuzzy Overlap technique. Through this analysis, we aimed to understand how our method performs in comparison to other existing similarity functions, particularly in terms of accuracy and efficiency.

Our tests, conducted using a range of datasets, showed that the standalone Fuzzy Overlap method achieved the best accuracy at 85.09%, with a slight decrease to 85.01% when combined with the Q-gram filter. These results highlight the accuracy of our approximated Q-gram filter and affirm the correctness of our mathematical approach.

An important finding of our study is the significant speed improvement observed when combining the Q-gram filter with Fuzzy Overlap. This combination was able to complete tasks in only 220 ms, much faster than the 13s:426ms needed by Fuzzy Overlap alone. This speed is especially notable because it brings the process into the realm of real-time capability, a critical factor for applications requiring immediate data processing. The α threshold parameter, which influences the number of potential matches considered in the second stage of Fuzzy Overlap, plays a key role in this efficiency.

Overall, our research provides useful insights into the use of similarity functions for large datasets, showing a promising balance between accuracy and speed. This balance is essential for practical applications like data analysis and integration, where real-time data processing can be crucial. Future research could focus on further optimizing these functions for even faster and more accurate real-time applications.

Acknowledgment

It was supported by SGS FEI UPCE 2024 and the Erasmus+ project: Project number: 2022-1-SK01-KA220-HED-000089149, Project title: Including EVERyone in GREEN

Data Analysis (EVERGREEN) funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the Slovak Academic Association for International Cooperation (SAAIC). Neither the European Union nor SAAIC can be held responsible for them.

References

1. Cohen, W.W., Ravikumar, P., Fienberg, S.E., et al.: A comparison of string distance metrics for name-matching tasks. In: IIWeb. vol. 3, pp. 73–78 (2003)
2. Deng, D., Kim, A., Madden, S., Stonebraker, M.: Silkmoth: An efficient method for finding related sets with maximum matching constraints. arXiv preprint arXiv:1704.04738 (2017)
3. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: A survey. IEEE Transactions on knowledge and data engineering 19(1), 1–16 (2007)
4. Gali, N., Mariescu-Istodor, R., Hostettler, D., Fränti, P.: Framework for syntactic string similarity measures. Expert Systems with Applications 129, 169–185 (2019)
5. Li, B.H., Liu, Y., Zhang, A.M., Wang, W.H., Wan, S.: A survey on blocking technology of entity resolution. Journal of Computer Science and Technology 35, 769–793 (2020)
6. Okazaki, N., Tsujii, J.: Simple and efficient algorithm for approximate dictionary matching. In: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). pp. 851–859 (2010)
7. Papadakis, G., Skoutas, D., Thanos, E., Palpanas, T.: Blocking and filtering techniques for entity resolution: A survey. ACM Computing Surveys (CSUR) 53(2), 1–42 (2020)
8. Rozinek, O., Mareš, J.: The duality of similarity and metric spaces. Applied Sciences 11(4) (2021), https://www.mdpi.com/2076-3417/11/4/1910
9. Ukkonen, E.: Approximate string-matching with q-grams and maximal matches. Theoretical computer science 92(1), 191–211 (1992)
10. Wang, J., Li, G., Fe, J.: Fast-join: An efficient method for fuzzy token matching based string similarity join. In: 2011 IEEE 27th International Conference on Data Engineering. pp. 458–469. IEEE (2011)
11. Wang, J., Lin, C., Zaniolo, C.: Mf-join: Efficient fuzzy string similarity join with multi-level filtering. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE). pp. 386–397. IEEE (2019)
12. Yang, Z., Yu, J., Kitsuregawa, M.: Fast algorithms for top-k approximate string matching. In: Twenty-Fourth AAAI Conference on Artificial Intelligence (2010)
13. Zeakis, A., Skoutas, D., Sacharidis, D., Papapetrou, O., Koubarakis, M.: Tokenjoin: Efficient filtering for set similarity join with maximumweighted bipartite matching. Proceedings of the VLDB Endowment 16(4), 790–802 (2022)