Jurnal Ilmiah GIGA Volume 28 (2) November 2025 Halaman 68-76 (p) ISSN 1410-8682
http://dx.doi.org/10.47313/jig.v28i2.4264 (e) ISSN 2621-9239

Fuzzy Matching Strategy Based on Specific Fields for Modified Data Leak Validation: An Experimental Study of Many-to-One Matching

Strategi Pencocokan Fuzzy Berdasarkan Specific-Field Untuk Validasi Kebocoran Data Termodifikasi: Studi Eksperimen *Many-To-One Matching*

Fadlilah Izzatus Sabila[1*], Catur Adi Nugroho[2], Fauziah[3]

[1,2,3] Program Studi Magister Teknologi Informasi, Fakultas Teknologi Komunikasi dan Informatika, Universitas Nasional, Jl. Sawo Manila No.61, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta, 12520, Indonesia

* Corresponding author: fadlilahizzatussabila.2024@student.unas.ac.id

(*Received*: 02-10-2025; *Revised*: 21-11-2025; *Accepted*: 30-11-2025)

Abstrak. Kebocoran data pribadi menjadi isu yang kian serius, terutama ketika data yang bocor telah dimodifikasi sebagian untuk menghindari pencocokan langsung dengan sumber asli. Penelitian ini mengembangkan pendekatan fuzzy berbasis pemetaan algoritma terhadap setiap atribut (*field-algorithm pairing*) serta skema pembobotan berdasarkan relevansi, untuk mendukung pencocokan satu ke banyak data (*many-to-one)* antara data bocor dan database asli. Empat algoritma digunakan: Levenshtein, Jaro-Winkler, Token Sort Ratio, dan Cosine Similarity, dipilih berdasarkan karakteristik semantik atribut. Eksperimen dilakukan pada 10.000 data sintetis dengan berbagai skenario modifikasi, mencakup data bersih, modifikasi ringan, dan modifikasi berat Hasil menunjukkan performa tinggi pada data bersih dan modifikasi ringan (F1-score 0,90–1,00), namun menurun signifikan pada modifikasi berat (F1-score 0,10–0,45). Pendekatan ini menawarkan solusi ringan namun efektif untuk tahap awal verifikasi identitas dalam investigasi kebocoran data, serta membuka peluang pengembangan lebih lanjut melalui kombinasi algoritma dan penyesuaian ambang pencocokan secara adaptif.

Kata kunci: validasi data bocor, pencocokan *fuzzy*, *string matching*, pemetaan algoritma tiap field.

Abstract. Personal data leakage is becoming an increasingly serious issue, especially when the leaked data has been partially modified to avoid direct matching with the original source. This study develops a fuzzy approach based on algorithmic mapping of each attribute (field-algorithm pairing) as well as a weighting scheme based on relevance, to support a many-to-one data match between the leaked data and the original database. Four algorithms are used: Levenshtein, Jaro-Winkler, Token Sort Ratio, and Cosine Similarity, selected based on the semantic characteristics of the attributes. Experiments were conducted on 10,000 synthetic data with various modification scenarios, including clean data, light modification, and weight modification Results showed high performance in both clean data and light modification (F1-score 0.90–1.00), but significantly decreased in heavy modification (F1-score 0.10–0.45). This approach offers a lightweight yet effective solution for the early stages of identity verification in data leak investigations, as well as opening up opportunities for further development through a combination of algorithms and adaptive adjustment of matching thresholds.

Keywords: leaked data validation, fuzzy matching, string matching, algorithm mapping of each field.

INTRODUCTION

Validating suspected leaked data is an essential aspect of cyber security, especially in the context of digital forensic investigation and privacy protection. The main challenge arises from data transformation by perpetrators or systems—such as partial masking, character substitution, or rewriting—which causes the data to no longer match the original source directly. Although much of the latest research focuses on complex machine learning approaches for leak detection, traditional fuzzy matching algorithms like Levenshtein, Jaro-Winkler, Token Sort Ratio, and Cosine Similarity offer a lighter and more computationally efficient alternative.

Fuzzy matching becomes crucial because leaked data generally undergoes partial modification, requiring matching methods to be tolerant of variations. In the context of one-to-many matching, each leaked entry needs to be compared against all entries in the original database to detect possibilities [1]. This study does not compare algorithms uniformly but uses a field-algorithm pairing approach [2], assigning algorithms according to attribute characteristics (for example, Levenshtein for NIK/phone numbers, Token Sort Ratio for names, Cosine Similarity for addresses, Jaro-Winkler for categorical attributes). This is based on the assumption that errors and data transformations are field-dependent. Therefore, this research examines the effectiveness and efficiency of a fuzzy matching strategy by combining the most appropriate algorithm for each attribute in a many-to-one matching scenario against modified leaked data.

Previous studies have addressed string matching algorithms in a variety of contexts, including security, data processing, and deduplication. Zhang [3] reviews a number of classic algorithms such as Knuth-Morris-Pratt (KMP), Boyer-Moore, and BNDM, and highlights the balance between algorithmic efficiency and ease of implementation. Gupta et al. [4] emphasized the importance of efficiency in intrusion detection systems that work in real-time. Furthermore, Van Gennip et al. [5] proposed an unsupervised record matching approach based on soft TF-IDF and n-gram, which has been shown to be effective in handling incomplete data and containing typos.

Cutting-edge research by Shu et al. [6], [7]developed a transformative approach to data leak detection, relying on an n-gram-based sequence alignment algorithm. This approach is able to overcome the limitations of the intersection set method, especially in detecting partial leaks and modifying data structures. Another advantage lies in the application of the concepts of comparable sampling and sampling-oblivious alignment, which allow the system to maintain a representation of semantic similarity even if the data is shrinked or inserted. Additionally, the implementation of these algorithms in parallel on the GPU architecture allows for the achievement of high throughput, making it relevant for the needs of large-scale organizations. In addition, other studies highlight the performance of classical algorithms which shows that algorithms play a guiding role in future research in string-search algorithm to improve the average performance of the algorithm and reduce resource consumption [3].

In terms of algorithmic efficiency, the contributions of Deng et al. [8] and Duyster & Kociumaka [9] show that techniques such as edit distance, trie structure, and run-length encoding are able to speed up matching in large-scale data scenarios. Andrzejewski et al. [10] also highlight the importance of selecting algorithms based on the types of attributes processed, such as names, addresses, and emails.

On the theoretical side, Charalampopoulos et al. [11] introduced a meta-algorithmic framework for approximate pattern matching based on edit distance and Hamming distance. They show that the structure of the occurrence of fuzzy patterns can be modeled in a loose periodicity framework and can be mathematically limited in the number of occurrences, even when the data is in compressed form.

Spesific research related to fuzzy matching conducted by Kaufman & Klevs [12] introduced an adaptive solution for fuzzy matching cases when there is only one "noisy"

identifier field—very useful for many-to-one matching strategies and modified data validation. Hosseini et al. [13] introducing a deep-learning approach to fuzzy matching (candidate ranking), is useful if you want to compare the traditional field-specific fuzzy matching method with the learning approach. Rudwan et al. [14] discusses information leakage in the "fuzzy" scheme (relevant for risk evaluation and validation of data leakage when applying fuzzy matching/obfuscated distances) very useful for data leakage validation section.

This study aims to fill this gap by developing and evaluating the fuzzy matching specific-field approach, through algorithm mapping strategies based on field characteristics and heuristic attribute weighting. An evaluation was carried out on modified synthetic data in three scenarios, to measure effectiveness (precision, recall, F1-score) and computational efficiency (time and memory), as well as resilience to data transformation.

METHOD

Dataset

This research utilizes 10,000 synthetic personal data entries as the basis, from which leaked data samples of 5, 10, 15, and 20 rows are generated under three scenarios: clean, lightly modified (4 columns), and heavily modified (7 columns).

TABLE 1. Data modification criteria

Lightly Modifictaion	
Field	**Rationale for Implementation**
Name	Change a character randomly (typo)
Email	Swap two adjacent character sequences
Phone Number	Change a random two digits with a symbol (*)
National ID Number	Change a random three-digit with a symbol (*)
Heavily Modifictaion	
Field	**Rationale for Implementation**
Name	Change the entire name with a random code (pseudonymization)
National ID Number	Change a random 10 digits with a symbol (*)
Phone Number	Change the 12-digit random string (after fromat 08)
Address	Changing to the name of the general district/city
Email	Change the entire username and domain section randomly
Driving License Number	Replace the entire format
Passport Number	Replace the entire format

The primary approach is field-algorithm pairing, which is the semantic mapping of attributes to the appropriate fuzzy matching algorithm. The primary approach is field-algorithm pairing, which is the semantic mapping of attributes to the appropriate fuzzy matching algorithm. For instance, Levenshtein is used for NIK (National Identity Number) and phone numbers, Token Sort Ratio for names, Cosine Similarity for addresses, and Jaro-Winkler for categorical attributes. Matching is performed using a many-to-one scheme, followed by score aggregation with heuristic weighting between fields.

Experiments

The experiments were conducted to test the performance of fuzzy matching across three data modification scenarios (clean, light, heavy) and four leaked data sample sizes (n = 5, 10, 15, 20). Each combination of scenario and size was tested 5 times to avoid bias resulting from random sample selection. Figure 1 shows the overall research workflow, starting from the generation of leaked data, field-by-field algorithm mapping, many-to-one matching, score weighting, up to the evaluation of results.

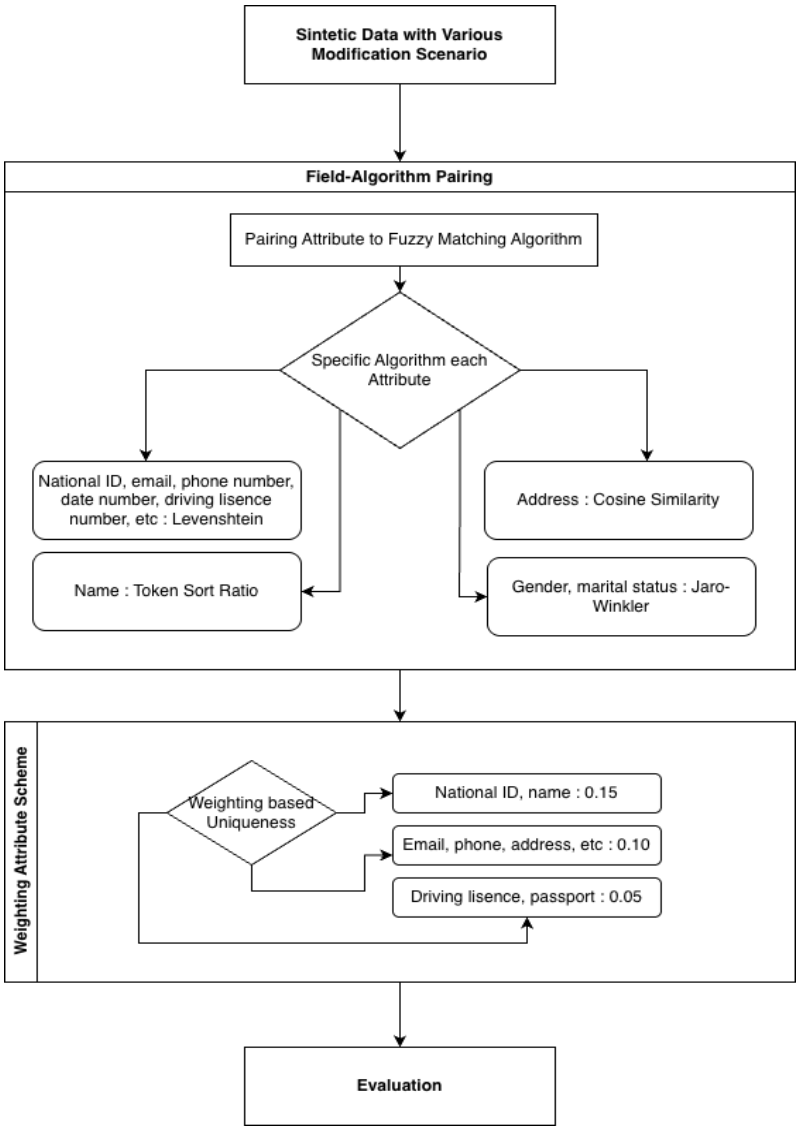

FIGURE 1. Research workflow

Field-Algorithm Pairing

The proposed field-algorithm pairing strategy aims to improve data matching effectiveness by mapping each attribute (field) to one of the four fuzzy matching algorithms that is most semantically and structurally suitable. The algorithm selection is based on the common characteristics of data transformation frequently observed in each field, such as typos, shuffling, and partial substitution. Table 2 shows the rationale for the algorithm selection.

TABLE 2. Field-algorithm pairing

Field	Algorithm	Rationale for Selection
Name	Token Sort Ratio	Detects changes in word order and minor typos.
National ID Number	Levenshtein	Handles changes in numerical digits and symbol substitution.
Date of Birth	Levenshtein	Consistent numerical format, sensitive to edits.
Phone Number	Levenshtein	Prone to digit changes and character substitution.
Address	Cosine Similarity	Effective vector representation for long text.
Driving License Number	Levenshtein	Numerical structure similar to NIK.
Passport Number	Levenshtein	Combination of letters and numbers suitable for edit distance.
Gender	Jaro-Winkler	Short string, tolerant of minor variations.
Email	Levenshtein	Often experiences character swaps or partial masking.
Marital Status	Jaro-Winkler	Limited values and short variations are suitable for Jaro-Winkler.

Matching Scheme

Matching was performed by running each algorithm against all designed data modification scenarios using a many-to-one scheme. The aggregate score was calculated based on attribute weighting, and the results are evaluated using matching accuracy metrics, resilience, execution time, memory usage, precision, recall, and F1-score.

Attribute Weighting Scheme

Inter-field matching utilizes a heuristic weighting scheme that represents the relative contribution of each attribute to the total similarity score. Weights were determined based on semantic considerations and the distinctiveness of the attributes in identifying individuals [15]. For illustration, unique attributes like NIK or email are given a larger weight compared to general attributes like gender. Table 2 presents the weights used in this study, with a total accumulation of 1.00.

TABLE 3. Attribute weighting

Field	Weight	Rationale for Implementation
Name	0.15	Informative yet prone to typos and word order reversals
National ID Number	0.15	Unique and rarely changing identification; sensitive to noise or masking
Email	0.15	Generally unique per individual, but at risk of masking or character swaps
Date of Birth	0.10	Potentially informative but often subject to digit blurring
Phone Number	0.10	Prone to digit changes and character substitution.
Address	0.10	Descriptive and informative, although the format can vary
Marital Status	0.10	Limited grades and not too discriminatory between individuals
Gender	0.10	Predictable and uninformative binary categories for disambiguation
Driving License Number	0.05	Similar to NIK, but less common as an identity key
Passport Number	0.05	Alphanumeric structures are unique, but they are not always available to all individuals

Evaluation

The evaluation was conducted based on two main aspects: (1) matching accuracy, measured using the metrics of precision, recall, and F1-score, and (2) computational efficiency, covering execution time (runtime) and maximum memory usage when matching 20 entries against the 10,000 original data entries.

All testing was performed in a controlled environment with consistent hardware and software configurations. The results were expressed in tables and graphical visualizations to compare the performance of each algorithm across every scenario.

RESULT AND DISCUSSION

Matching Analysis

The experiment was conducted by matching leaked data (n = 5, 10, 15, and 20 entries) against 10,000 original data entries in each modification scenario (clean, light modification, and heavy modification). The similarity score was calculated based on the aggregation of scores per field using the attribute weighting scheme.

TABLE 4. Average similarity score per scenario (n = 10 entries)

Scenario	Number of Entries	Average Score	Standard Deviation
Clean	10	1.0	9.93×10^{-17}
Light Modification	10	0.942	0.002
Heavy Modification	10	0.495	0.013

In clean scenario, all attributes are still identical to the original data. The system generated an F1-score of 1.00 on all sample sizes, with an average similarity score of 1.00 and a standard deviation of 9.93×10^{-17}. The value indicates perfect stability and full consistency between runs, which corresponds to data conditions without modification. In the light modification scenario, although minor changes were present, the system still maintained a

high consistency of similarity scores. However, in the heavy modification scenario, there was a significant drop, indicating the challenges in dealing with complex data transformations.

In the lightly modified scenario, the system still achieves high performance with an average similarity score in the range of 0.94–0.97, and an F1-score of 0.93–1.00 in most sample sizes. The decline in performance at certain sample sizes is due to modifications that affect high-weighted attributes, such as names or NIKs, which directly impact the aggregate score and cause some pairs to not exceed the threshold of match.

This heavily modified scenario shows a more significant decline in performance, with an average similarity score of only 0.48–0.49 and an F1 score of 0.10–0.45. This is because most attributes undergo major structural changes, such as pseudonymization of names, randomization of NIK digits, and substitution of location on addresses. The loss of core information causes the fuzzy algorithm to lose an adequate comparative base, resulting in a substantial decrease in match rates.

Standard deviation measurements were performed to assess performance consistency in the five tests of each scenario. The results are as follows: Net Data with a standard deviation of 9.93×10^{-17} indicates perfect stability, Mild Modification with a standard deviation of 0.002 indicates very small variation and consistent performance, and Heavy Modification with a standard deviation of 0.013, indicates slight fluctuations due to random variations in the attribute modification process. A low standard deviation value indicates that the matching method has a good level of reliability, even in scenarios with different levels of data corruption.

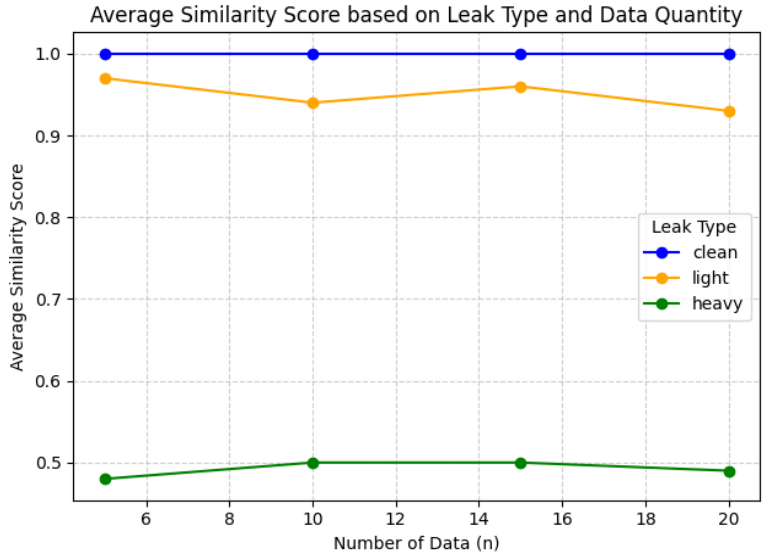

FIGURE 2. Average similarity score across three data scenarios (n = 10 entries)

Overall, the pattern apparent in Figure 1 indicates that the higher the level of data modification or leakage, the more the average similarity score tends to decrease. This suggests that the algorithm's resilience to data changes has limits, and its performance heavily relies on the quality of the input. The simultaneous use of the mean (average) and standard deviation is a standard practice in scientific research aimed at ensuring that the reported results are not merely coincidental but have a strong statistical basis.

Precision, recall, dan F1-Score

TABLE 5. Classification metrics for light modification scenario (threshold 0.90)

Number of Leaked Entries	Precision	Recall	F1-Score
5	1.00	1.00	1.00
10	0.90	0.90	0.90
15	1.00	1.00	1.00
20	0.95	0.95	0.95

Table 5 indicates that the classification metric results in the light modification scenario vary according to the number of leaked entries, with data sizes of 5 and 15 having the best results, valued at 1.00, followed by 20 entries with a value of 0.95, and 10 entries with a value of 0.90. The determination of a threshold of 0.90 for the light modification scenario is based on empirical analysis to balance between precision and recall [12]. Using a threshold that is too high (>0.95) risks increasing false negatives (failing to detect matching data), while a threshold that is too low can increase false positives. A value of 0.90 was chosen because, at this level, the system could ble to accommodate minor variations such as one or two-character typos without incorrectly classifying unmatched data.

TABLE 6. Classification metrics for heavy modification scenario (threshold 0.48)

Number of Leaked Entries	Precision	Recall	F1-Score
5	0.20	0.20	0.20
10	0.10	0.10	0.10
15	0.53	0.53	0.53
20	0.45	0.45	0.45

Table 6 shows that under heavy modification, the data size of 15 yields the highest classification metric value at 0.53, followed by the data size of 20 with a value of 0.45, the data size of 5 with a value of 0.20, and the data size of 10 with a value of 0.10. In the heavy modification scenario, lowering the threshold to 0.48 is a necessary adaptive step. Extensively modified data will inherently result in lower similarity scores. If the threshold of 0.90 were maintained, almost all true matching pairs would be rejected (false negatives), leading to a very low recall. Dynamically adjusting the threshold based on the characteristics of the data being analyzed is a common strategy for optimizing the performance of matching models. For further development, an adaptive thresholding approach, where the system automatically learns the optimal threshold based on the distribution of similarity scores from data samples, can be implemented to avoid manual determination and improve model generalization.

TABLE 7. Runtime and memory evaluation each algorithm

ALgorithm	Execution Time (s)	Memory Usage (MB)
Levenshtein	0.296	293.554
Jaro-Winkler	2.551	293.562
Token Sort Ratio	1.237	293.566
Cosine Similarity	163.201	293.718

Based on the test results in Table 7, the Levenshtein algorithm is proven to be the most computationally efficient, with the fastest execution time of 0.296 seconds and the lowest memory usage (293.554 MB). This makes Levenshtein moderately suitable for numerical or structured fields with limited variation, such as NIK (ID number), date of birth, or phone number, where errors are typically just a typo or a single character shift. Although Jaro-Winkler and Token Sort Ratio also showed relatively efficient performance, both still had a higher execution time compared to Levenshtein. Jaro-Winkler excels for short fields like gender or marital status, while Token Sort Ratio is more robust against differences in word order and is suitable for names. Meanwhile, Cosine Similarity has the highest computational load, with an execution time reaching 163.201 seconds. However, this algorithm remains relevant for long, unstructured text fields such as addresses, as it can capture semantic similarity despite variations in wording. Therefore, the utilization of Levenshtein as the default algorithm for short text and numerical fields is considered the most optimal in terms of runtime and memory efficiency, particularly in large-scale matching scenarios.

Comparison with Baseline

Prior to the execution of this experiment, the authors conducted a series of preliminary tests using a baseline approach, namely fuzzy matching with a single algorithm applied uniformly across all fields (uniform matching). This approach utilized five individual fuzzy

algorithms: Levenshtein, Jaro-Winkler, Token Sort Ratio, Cosine Similarity, and Jaccard, without considering the specific characteristics of each attribute. The experiments were performed with a dataset structure and modification scenarios identical to those in this study. The results indicated that all algorithms had high precision (approaching 1.0) but very low recall (generally < 0.002), resulting in an F1-score < 0.004 in the modified leaked data scenario. This decline was attributed to the inability of the uniform approach to handle the semantic and structural variations that occurred across different fields, such as changes in name order or substitutions in numerical fields. Conversely, the field-by-field algorithm pairing approach used in this research showed a significant improvement in recall and F1-score, especially in scenarios with minor modifications. The field–algorithm pairing approach has proven to be effective in maintaining matching accuracy under various data conditions. The selection of algorithms based on the semantic characteristics of each attribute contributes to increased accuracy, especially on minor modifications. Token Sort Ratio provides the best results for name attributes because it is able to recognize changes in the sequence of tokens, Levenshtein is effective on numerical attributes that are susceptible to digit modification, Cosine Similarity shows the best performance for long text such as addresses, and Jaro-Winkler is optimal for short categorical attributes. Thus, this strategy provides better flexibility and resilience than a uniform approach that uses a single algorithm for the entire field, although this study does not explicitly compare the two.

CONCLUSION

This research evaluates the effectiveness of a specific-field fuzzy matching approach for the verification of modified leaked data, using a field-algorithm pairing and many-to-one matching strategy. This allows the system to handle partial inconsistencies between attributes in a flexible and contextual manner. The experimental results consistently show that the strategy of pairing algorithms with fields provides better performance compared to the approach of using one algorithm for all fields, as it can adapt the matching method to the semantic and data format characteristics of each attribute. The system demonstrated high performance on clean data and data with minor modifications (F1-score 0.90–1.00), but a significant decline was observed with heavy modifications (F1-score 0.10–0.45). An evaluation of algorithm efficiency indicated a trade-off between accuracy and computational resources: Levenshtein is efficient in time and memory, while Cosine Similarity excels for long text fields, albeit with a greater computational load. The heuristic weighting scheme across fields was proven to increase the accuracy of the aggregation score, particularly on partially modified data, by assigning proportional contributions based on the attribute's relevance to identity. The optimal results were achieved at a sample size of n=15, although this finding is empirical and influenced by the variation in random sampling data distribution. Overall, this fuzzy matching approach based on field-algorithm pairing and contextual weighting is promising as a lightweight and effective solution for initial verification in data leak investigations. However, for real-time production cases, further optimization or integration with a hybrid system is required.

REFERENCES

[1] H. Kasyap, U. I. Atmaca, C. Maple, G. Cormode, and J. He, “Privacy-preserving Fuzzy Name Matching for Sharing Financial Intelligence,” Nov. 2024, [Online]. Available: http://arxiv.org/abs/2407.19979

[2] B. P. K., “Fuzzy Approach to Record Linkages,” *Preprint ArXiv:2402.03464*, 2024, doi: 10.48550.

[3] Z. Zhang, “Review on String-Matching Algorithm,” *SHS Web of Conferences*, vol. 144, p. 03018, 2022, doi: 10.1051/shsconf/202214403018.

[4] V. Gupta, M. Singh, and V. K. Bhalla, "Pattern matching algorithms for intrusion detection and prevention system: A comparative analysis," *Proceedings of the 2014 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2014*, pp. 50–54, Nov. 2014, doi: 10.1109/ICACCI.2014.6968595.

[5] Y. van Gennip, B. Hunter, A. Ma, D. Moyer, R. de Vera, and A. L. Bertozzi, "Unsupervised record matching with noisy and incomplete data," Int J Data Sci Anal, vol. 6, no. 2, pp. 109–129, Sep. 2018, doi: 10.1007/s41060-018-0129-7.

[6] X. Shu, J. Zhang, D. D. Yao, and W. C. Feng, "Rapid screening of transformed data leaks with efficient algorithms and parallel computing," in CODASPY 2015 - Proceedings of the 5th ACM Conference on Data and Application Security and Privacy, Association for Computing Machinery, Mar. 2015, pp. 147–149. doi: 10.1145/2699026.2699130.

[7] X. Shu, J. Zhang, D. Yao, and W. C. Feng, "Fast detection of transformed data leaks," IEEE Transactions on Information Forensics and Security, vol. 11, no. 3, pp. 528–542, Mar. 2016, doi: 10.1109/TIFS.2015.2503271.

[8] D. Deng, G. Li, H. Wen, H. V. Jagadish, and J. Feng, "META: An efficient matching-based method for error-tolerant autocompletion," Proceedings of the VLDB Endowment, vol. 9, no. 10, pp. 828–839, Jun. 2016, doi: 10.14778/2977797.2977808; Taxonomy: ACM-PUBTYPE; PAGEGROUP: STRING: PUBLICATION.

[9] A. Duyster and T. Kociumaka, "Logarithmic-Time Internal Pattern Matching Queries in Compressed and Dynamic Texts," Mar. 2025, [Online]. Available: http://arxiv.org/abs/2503.03488

[10] W. Andrzejeski, B. Bebel, P. Boinski, M. Sienkiewicz, and R. Wrembel, "Text Similarity Measures in A Data Deduplication Pipeline for Customers Records," in Proceedings of the 25th International Workshop on Design, Optimization, Languages and Analytical Processing of Big Data (DOLAP), Mar. 2023, pp. 33–42. [Online]. Available: https://ceur-ws.org/Vol-3369/

[11] P. Charalampopoulos, T. Kociumaka, and P. Wellnitz, "Faster approximate pattern matching: A unified approach," in Proceedings - Annual IEEE Symposium on Foundations of Computer Science, FOCS, IEEE Computer Society, Nov. 2020, pp. 978–989. doi: 10.1109/FOCS46700.2020.00095.

[12] A. R. Kaufman and A. Klevs, "Adaptive Fuzzy String Matching: How to Merge Datasets with Only One (Messy) Identifying Field," Political Analysis, vol. 30, no. 4, pp. 590–596, 2022, doi: DOI: 10.1017/pan. 2021.38.

[13] K. Hosseini, F. Nanni, and M. Coll Ardanuy, "DeezyMatch: A Flexible Deep Learning Approach to Fuzzy String Matching," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Q. Liu and D. Schlangen, Eds., Online: Association for Computational Linguistics, Oct. 2020, pp. 62–69. doi: 10.18653/v1/2020.emnlp-demos.9.

[14] M. S. M. Rudwan and J. V. Fonou-Dombeu, "Hybridizing Fuzzy String Matching and Machine Learning for Improved Ontology Alignment," Future Internet, vol. 15, no. 7, 2023, doi: 10.3390/fi15070229.

[15] Y. Zhang, S. Xu, M. Zheng, and X. Li, "Field Weights Computation for Probabilistic Record Linkage in Presence of Missing Data," *Intern J Pattern Recognit Artif Intell*, vol. 34, no. 14, Dec. 2020, doi: 10.1142/S0218001420590466.