Scientific Journal of Impact Factor (SJIF): 5.71
e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 7, Issue 12, December -2020

biHash: A Private Approach for Data Fusion

Ms. Priyanshi Mulwani[1], Prof. Vijay Shelake[2]

[1]Department of Computer Engineering, YTCEM, Mumbai, India

[2]Department of Computer Engineering, YTCEM, Mumbai, India

Abstract - *Now-a-days, the huge amount of data is distributed across multiple providers. There is a need to combine such scattered data for mining tasks. Thus, data fusion plays an important role in combining records from various sources However, security is a major concern while performing data fusion. It is thus the concern and responsibility of organizations and corporate companies to keep the private data of customers secure during data fusion. This paper presents a private approach for data fusion with biHash algorithm and its analysis with Levenshtein operations and exact method for comparison of records.*

Keywords - *security, data fusion, biHash, Levensthein, SHA3_512*

I. INTRODUCTION

In this technological era almost every financial, governmental, medical and social transaction is logged and stored into a database. Thus, each organization maintains large amount of private information for every customer, the need for promising privacy is more demanding than ever. This need becomes more intensive when it comes to cases where two organizations decide to combine their data, for statistical purposes. Today's applications incline towards usage of confidential information for achieving better performance with their services. There is a need to protect data because of untrusted third parties, data privacy of customer is not conceded but the actions on it would be compatible. Data fusion is the task of merging instances from two or more sources to generate new source with additional knowledge representation. The computation with edit distance is found to be appropriate for secure data fusion [1][2][3][4][5][6].

II. BACKGROUND

The merging of information from multiple sources to create sophisticated models can hamper the privacy of databecause of bringing together heterogeneous data sources. The different existing analytical approaches do not prove to be ensuring security and provide appropriate metrics for evaluation. Understanding these potential challenges, along with the complexities surrounding different kinds of heterogeneity in the data is the key to thresearch work for developing viable data fusion approaches. The different mechanisms for secure data fusion are discussed as follows:

IraklisLeontiadis, RefikMolva, M.J. Chorley and G.B. Colombo [5] suggested a similarity detection method for analysis of data to preserve privacy. They analyzed an algorithm for similarity detection, known as the cosine similarity. They adopted obfuscation mechanisms with cosine similarity for achieving similarity computation solution with privacy preserving. They offered a protocol for similarity detection. This mechanism focuses on mapping of user's data into various vectors. Geometrical transformations are applied preserving the angle for pair of vectors. It gives assurance to maintain the confidentiality of the content.

Frank Breitinger, Georg Ziroff, Steffen Lange, Harald Baier[6] have suggested a hashing approach for similarity detection called saHash algorithm. This statistical analysis hashing works on creating four sub-hash functions. The four sub-hash values are merged together to generate the final hash value. saHash permits the detection of "small" variations between files. The saHash method has three advantages. Mentioning the first, it is closely as fast as SHA-1. Second, it produces approximately fixed-size hash values. Third, similarity is defined in terms of the Levenshtein distance. saHash is robust when small deviations have been made over a file. One disadvantage is it fixes the computation on similarity of inputs which are of equal sizes.

Benjamin Fabian and Tom Göthling [7] have suggested privacy preserving data warehousing techniques. They have chosen various anonymization concepts and techniques with respect to data warehouse context. The main aim is to estimate the applicability of database privacy concepts to current trend of data warehousing. Appropriate protection mechanisms are selected and possible threats are identified by data publishers. The addition of new sources, the barring of sources, data handling operations, and data sanitization were recognized as privacy affecting process steps. Delete operations are not handled by incremental l-diversity. The statistical properties of the data and the dissemination of delicate attribute values impact the data utility.

Juan Manuel Dodero, Mercedes Rodriguez-Garcia and Enrico Motta [8] proposed classification framework in linked data mashup architectures for privacy conserving data publishing. A data hub is formed by the mashup of microdata sources. By doing so privacy preservation anonymity requirements must be fulfilled which proves to be obstacle for data analysts

to point out sensitive information of the sources. On this baseline, an approach was designed for privacy-preserving linked data mashups which focuses the importance of privacy-preserving linked data publishing architectures.
J.O Abe,Burak and Berk Üstündağ [9] have proposed A Data as a Service (DaaS) Model for GPU-based data analytics. To give a real time data analysis from various domains a model of GPU system for Data As A Service (DaaS) was presented. A customized and profitable system for DaaS on GPU was initiated by using a modeled layer to define a learning protocol. Their contribution included a model system that is scalable and functional for DaaS through a GPU based, machine learning enabled analytics model and conducted preliminaries experiment on GPU enabled system to test the pre-processing system of the DaaS model. Table 1 shows the comparison of different mechanisms for secure data fusion.

Table 1. Comparison of Different Mechanisms for Secure Data Fusion

Sr. No	Author Name	Year	Brief Description	Benefits	Limitations
1.	IraklisLeontiadis, RefikMolva, M.J. Chorley and G.B.Colombo [5]	2013	Performs analysis of cosine similarity detection algorithm along with obfuscation mechanisms in order to accomplish a privacy preserving similarity computation solution.	Mapping of user data into vectors. Maintains the confidentiality of the content. Application of geometrical transformations which preserves the angle between pair of vectors.	Mathematical background needed. Complicated calculations involved.
2.	Frank Breitinger, Georg Ziroff, Steffen Lange, Harald Baier [6]	2014	Utilizes similarity hashing method called saHash which uses four sub hash functions.	Similarity is defined in terms of the Levenshtein distance. Nearly as fast as SHA-1. Fixed-size hash values.	Compute the similarity of inputs dealing with similar sizes.
3.	Benjamin Fabian and Tom Göthling [7]	2015	Estimate the applicability of database privacy notions with respect to data warehousing.	Recognize probable threats and choose appropriate protection mechanisms. Data manipulation actions and data sanitization were acknowledged as privacy affecting process steps.	Unable to handle delete operations. The statistical properties of the data, the dissemination of delicate attribute values impact data utility.
4.	Juan Manuel Dodero, Mercedes Rodriguez-Garcia and Enrico Motta [8]	2017	Suggested classification framework in linked data architecturesfor Privacy-Preserving Data Publishing.	It can decide on the distribution of control and partitioning of the dataset information models. It's an approach to engineer privacy-preserving linked data mashups.	The mashup of microdata sources to form a data hub must fulfil a set of privacy preservation anonymity requirements.
5.	J.O Abe,Burak and Berk Üstündağ [9]	2018	Provides overlay model of GPU system for Data As A Service (DaaS) to give a real-time data analysis of network data, customers, investors and users' data from the data centers or cloud system.	Custom, profitable system for DaaS on GPU. It is a scalable and functional model for DaaS through a GPU based Machine learning enabled analytics model.	High processing power machines needed.
6.	Xin Su, Kuan Fan and WenboShi [10]	2019	Satisfies k-Anonymous and differential privacy for secure data fusion.	It provides security to individual attributes in data mining purposes.	Some attributes can be disadvantageous for secure data fusion.

The secure distributed Data Fusion framework [10] satisfies K-Anonymous and differential privacy properties for security in data fusion. It can efficiently preserve information in data mining activities. The effort focuses on pointing towards the problem of privacy leaks in the procedure of data fusion. The authors present a privacy model for distributed

data fusion system with k-anonymous and non-interactive differential privacy. It introduces the concept of contribution in the process of data fusion dealing with the real situation of redundancy.

III.SECURE DATA FUSION

Data fusion [10] can be stated as a combination of multiple sources to obtain improved information with higher quality with precision and without hampering privacy of individuals. Considering 'Security' as the need of an hour, biHash algorithm is developed to ease the two parties owning their respective data by hashing the sensitive values
for data fusion. The Figure 1 shows the different components for secure data fusion. The different components of secure data fusion include:

- **Data Cleaning:**

  Pre-processing data, like cleaning and standardization increases data fusion accuracy. The cleaning of strings across the data sources performed by removing unwanted tokens, whitespace and brackets. In our datasets like Yelp.csv, cleaning was done by removing extra white spaces and " ".

- **Sampling and Indexing:**

  Datasets are huge in nature. For research purposes samples of these huge datasets are chosen at random. The main purpose of indexing is to make pairs of records for the given source datasets. Various indexing approaches are available like block, sorted neighborhood and full indexing. Candidate links or candidate matches are generated as the result of indexing. Here, block indexing is used, it is applied on attribute 'name' from the datasets which is used further for record linkage purposes.

- **Hashing and Data Fusion:**
  For security issues, few fields are hashed in both datasets by using biHash function. In this approach a string is divided into two sub-strings and separate hash function is applied to each sub-string and then concatenated. This bihashed concatenated final string is written to new file, which is further used for data fusion purposes.
  A set of informative, discriminating and independent features is important for a good classification of data values into similar and distinct pairs. Prior to data fusion, it consists of comparing and classification of data. Data fusion merges similar pairs and store in a single data source.

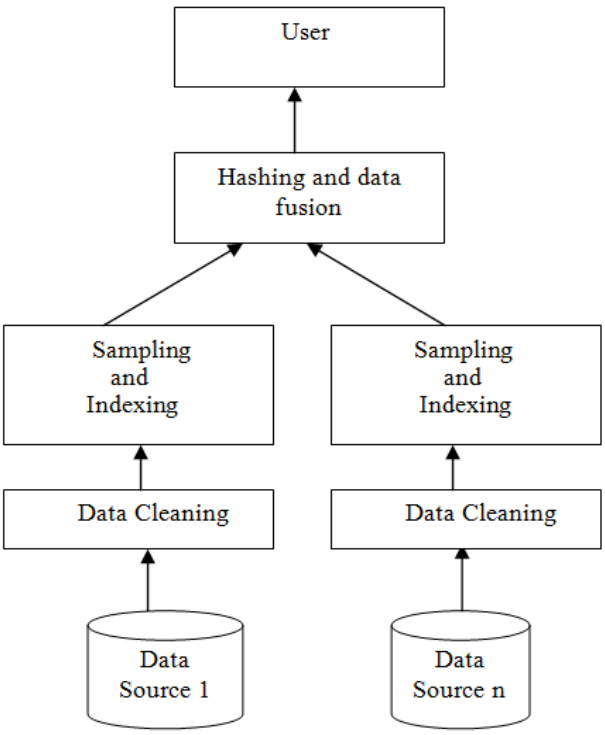

Fig.1. Proposed System

This section focuses on the biHash algorithm for data fusion. The algorithm evaluates byte-level similarity centered on the Levenshtein distance. Levenshtein distance is also called as edit distance method based o distance metrics. The Levenshteindistance[1] is a metric for strings to measure the difference between two sequences. It is the minimum number of single-character edits (i.e. insertions, deletions, or substitutions) required to change one word into the other, related to pairwise string alignments.
Following illustration shows how Levenshtein algorithm works:
String 1= “data”
String 2= “data”
In the above example it is clear that no transformations are needed as source and target strings are identical. Thus stating that the Levenshtein distance is 0 .
If String 1= "test"
String 2= “team"
The Levenshtein distance is 2 stating that two substitutions re needed to change "test” to "team".
Exact method is used to compare records in an exact way. The similarity is 1 in case of agreement and 0 otherwise.
The biHash algorithm uses two independent sub-hash functions. Each sub-hash function produces its own hash value. The final string is produced by concatenating the two sub-hash values. The first sub-string is generated using SHA3_512 hash algorithm [2] which produces a hash string of 512 bits and the other sub-string using SHA256 produces a 256-bit (32 bytes) hash value. It's usually represented as a hexadecimal number of 64 digits [3]. The security for data fusion contains the following procedure:

- Step I: Calculate length of base string
  For example : Base string = Messina

Length of above Base string = 7

- Step II: Find the division limit

Division Limit = length/2= 3
Thus given String ‘Messina’ gets its division length of 3 that is first sub string consists of first three characters and second sub string consists of last four characters.

- Step III: Find first sub string
  Thus first sub string = Mes (first three characters)

- Step IV: Find second sub string
  Thus second sub string = sina (last four characters)
  In this way, base String is divided into two parts: Substring 1= Mes [3]

Substring 2= sina [4]

-
- Step V: Hash the first sub string by SHA3_512 algorithm producing 512- bit hash value.

For above example substring ‘Mes’ is hashed as:
‘15e43f761389a0be18b1d9207139765910eee7f815f4616346251330de9541fcefbacd32590f512a7138f14c78f24865da2099acafc8f016f9f8c5a7a112e66’ producing 512 bit hash value.

- Step VI: Hash the second sub hash by SHA256 producing a 256-bit hash value.

For above example substring ‘sina’ is hashed as:
‘ef014ce3721b7f20d9c775b0e4457f422ab74b1b48c9136c63b97ab611fbf10e’ producing 256 bit hash value.

- Step VII: Concatenate hashed substring 1 and hashed substring 2 and store in a single string.

Finally, the string ‘Messina’ results into the following hashed output:
‘15e43f761389a0be18b1d9207139765910eee7f8a15f4616346251330de9541fcefbacd32590f512a7138f14c78f24865da2099acafc8f016f9f8c5a7a112e66ef014ce3721b7f20d9c775b0e4457f422ab74b1b48c9136c63b97ab611fbf10e’

IV. **DISCUSSION AND ANALYSIS**

In this work, biHash algorithm was executed on two datasets Yelp.csv (3000 records) and Tripadvisor.csv (1000 records) for private data fusion. The Yelp.csvconsist of 13 fields and Tripadvisor.csv consist of 12 fields. Data fusion was done for five similar fields namely name, address, locality, state and postal code. These fields were hashed by using biHash algorithm, ensuring security. The dictionary attack is not possible due to the use of secure hashing method. The machine specifications are as under: Intel Core i7-4710MQ CPU @2.50 GHz with 16 GB RAM on Windows platform and python language.
Figure 2 shows the analysis of biHash algorithm along with Levenshtein operations and exact method for comparison of records.

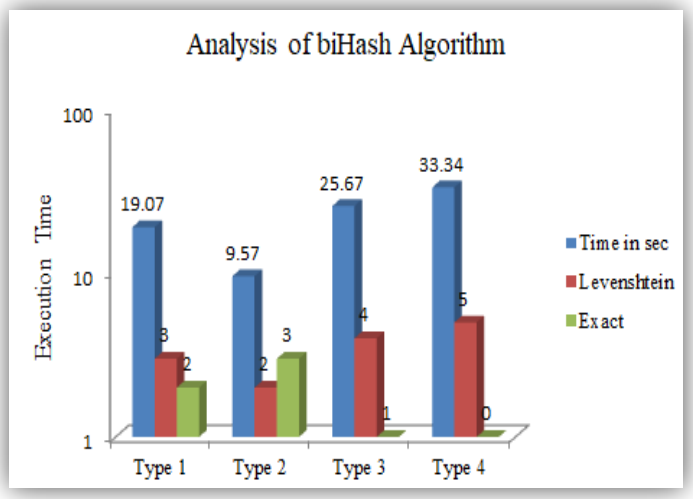

Fig. 2 Analysis of biHash algorithm

Above graph shows the difference in execution time of biHash algorithm when executed with attribute combinations using Levenshtein method and exact methods of comparison. In our research work, we have compared five attributes from the given datasets as stated above.
All the five attributes are hashed and then compared.The different type of combinations of attributes with exact and Levenshtein operations are as follows:

Type 1 with Levenshtein operations: name, address, city
Type 1 with Exact method: state, postal code

Type 2 with Levenshtein operations: name, address
Type 2 with Exact method: city, state, postalcode

Type 3 with Levenshtein operations: name, address, city, state
Type 3 with Exact method: postal code

Type 4 with Levenshtein operations: name, address, city, state, postal code
Type 4 with Exact method: NIL

Table 2 shows the various types of combinations used for analysis.

Table 2. Types of Combinations

Type/Operations	Number of attributes using Levenshtein operations	Number of attributes using Exact method
Type 1	3	2
Type 2	2	3
Type 3	4	1
Type 4	5	0

From figure 2 and table 2, it is clear that when biHash function is applied on above types, execution time is higher when number of attributes using Levenshtein operations is more compared to exact method.

V. CONCLUSION

In this paper, we have successfully implemented biHash algorithm on two datasets ensuring security. The principal advantages of biHash are its sectional design that permits it to be more secure and robust. The final hash value is expressed as concatenation of SHA3_512 and SHA256 hash values which proves to be more secure. Also, biHash has good runtime efficiency and is significantly faster than other hashing methods. By executing biHash algorithm on various types of combination it is observed that execution time is increasing with increase in number of Levenshtein operations.

REFERENCES

[1] Yi Mar Myint, "Edit distance computation with minimum number of edit operations in database management system and information retrieval," International Journal of Scientific and Research Publications, Volume 9, Issue 9, 2019.
[2] Meiliana Sumagita, Imam Riadi , "Analysis of secure hash algorithm(SHA) 512 for Encryption Process on Web Based Application," International Journal of Cyber-Security and Digital Forensics (IJCSDF), 7(4): 373-381, 2018.
[3] A. Gowthaman, S. Manickam, "Performance study of enhanced SHA-256 algorithm", International Journal of Applied Engineering Research, Volume 10, Number 4, pp. 10921-10932, 2015.
[4]Michael Gilleland,Merriam Park Software, "Levenshtein Distance in Three Flavours" https://people.cs.pitt.edu/~kirk/cs1501/Pruhs/Spring2006/assignments/editdistance/Levenshtein%20Distance.htm.
[5] IraklisLeontiadis, Melek O¨ Nen, RefikMolva, M.J. Chorley, G.B. Colombo, "Privacy preserving similarity detection for data Analysis", 2013 International Conference on Cloud and Green Computing, Karlsruhe, pp. 547-552, IEEE, 2013.
[6] Frank Breitinger, Georg Ziroff, Steffen Lange, Harald Baier, Similarity Hashing Based on Levenshtein Distances, IFIP International Conference on Digital Forensics DigitalForensics 2014: Advances in Digital Forensics X, pp. 133-147, 2014.
[7] Benjamin Fabian and Tom Göthling, "Privacy-preserving data warehousing", Int. J. Business Intelligence and Data Mining, Vol. 10, No. 4, 2015.
[8] Juan Manuel Dodero, Mercedes Rodriguez-Garcia and Enrico Motta, "Privacy-Preserving Data Publishing in Linked DataMashup Architectures", PrivOn@ISWC2017, 2017.
[9] John Olorunfemi Abe and Burak Berk Üstündaˇg, "A Data as a Service (DaaS) Model for GPU-based Data Analytics," ArXiv abs/1802.01639, 2018.
[10] Xin Su, Kuan Fan and WenboShi, "Privacy-preserving Distributed Data Fusion Based on Attribute Protection", IEEE Transactions on Industrial Informatics, vol. 15, no. 10, pp. 5765-5777, IEEE, 2019.

