Entity Relationship Modeling for Enterprise Data Space Construction Driven by a Dynamic Detecting Probe

Ye Tao[1(✉)], Shuaitong Guo[1], Ruichun Hou[2], Xiangqian Ding[2], and Dianhui Chu[3]

[1] College of Information Science and Technology, Qingdao University of Science and Technology, Qingdao, China
ye.tao@qust.edu.cn, forfree@mails.qust.edu.cn

[2] College of Information Science and Engineering, Ocean University of China, Qingdao, China
houruichun@ouc.edu.cn, dingxq1995@vip.sina.com

[3] School of Computer Science and Technology, Harbin Institute of Technology (Weihai), Weihai, China
chudh@hit.edu.cn

Abstract. To solve the problem of integrating and fusing scattered and heterogeneous data in the process of enterprise data space construction, we propose a novel entity association relationship modeling approach driven by dynamic detecting probes. By deploying acquisition units between the business logic layer and data access layer of different applications and dynamically collecting key information such as global data structure, related data and access logs, the entity association model for enterprise data space is constructed from three levels: schema, instance, and log. At the schema association level, a multidimensional similarity discrimination algorithm combined with semantic analysis is used to achieve the rapid fusion of similar entities; at the instance association level, a combination of feature vector-based similarity analysis and deep learning is used to complete the association matching of different entities for structured data such as numeric and character data and unstructured data such as long text data; at the log association level, the association between different entities and attributes is established by analyzing the equivalence relationships in the data access logs. In addition, to address the uncertainty problem in the association construction process, a fuzzy logic-based inference model is applied to obtain the final entity association construction scheme.

Keywords: Entity association · Data space · Fuzzy logic · Dynamic detecting probe

1 Introduction

In recent years, a low level of information sharing and a disconnect between information and business processes and applications have become common in enterprise business systems, which can easily lead to the formation of information silos within the enterprise

J. Xiong et al. (Eds.): MobiMedia 2021, LNICST 394, pp. 185–196, 2021.
https://doi.org/10.1007/978-3-030-89814-4_14

[1]. In particular, industrial software companies need substantial technical data support to deliver programmatic industrial processes and technologies, which requires not only a solution to information silos within the enterprise to achieve data sharing but also the integration of data with many different industrial enterprises [2, 3]. Therefore, to integrate internal or external data, some enterprises began to build data space systems, trying to integrate ERP, SCM, MES and other industrial software to eliminate information silos.

Building a data space system mainly requires the accurate establishment of associations between entities, which for this paper means integrating heterogeneous data from multiple source databases into a comprehensive enterprise data space through entity matching. Current research results mainly focus on discovering associations between entities or attributes through the semantic matching of dictionaries or semantic libraries, using data representation or content similarity judgments [4] to calculate the probability of similarity between data. Many of these methods have poor generalizability, slow response and low accuracy when attempting to discover the existence of associations from large amounts of data.

In this paper, we propose a new approach to discover entity association relationships in big data. First, this approach obtains business logic information and database data through dynamic probes deployed between the business logic layer and data access layer of different systems. Then, it portrays the similarity degree among entities in three dimensions, schema, instance and log, and gives the similarity values between entities in these different dimensions. Finally, based on the fuzzy logic inference method [5, 6], the similarity values between entities in different dimensions are converted into normalized values that can be uniformly measured to obtain the best matching results of entity association.

2 Related Work

In academic research, entity association is mainly divided into two types: schema matching [7] and instance analysis [8]. Schema matching extracts structural features from data sources as metadata and analyzes them to achieve association matching between data with fewer resources; matching based on instance analysis analyzes the data itself to obtain matching information, which usually consumes more resources but can obtain more accurate and comprehensive analysis results.

For schema matching academic research, in [9], He et al. used Structured Query Language (SQL) to extract features such as the name of the database, name of the schema, and type of column as metadata sets from each selected dataset. Then, they joined all metadata from each dataset into a metadata database. Finally, the correlation between the metadata was calculated by different methods to establish an association between the source data.

In academic research on instance analysis, the preprocessing that mines associations between data includes categorized data. For example, for the data conflict problem in data fusion, in [10], the conflict can be divided into two categories, uncertain conflict and contradictory conflict, and then the duplicate data of the same representation are fused, thus solving problems such as the possible conflict between different values for the same attribute.

Academics are also studying the integration of deep learning with logs. Mohanty et al. [11] cleaned the web log files collected by the IoT, built user profiles, saved similar information, and proposed a recommendation system based on rough fuzzy clustering to recommend e-commerce shopping sites to users. We offer a proposal for extracting the data association information in logs, this approach builds on the feature that logs contain association information between data.

The constantly increasing amount of data accumulating in the development of enterprises leads to an increasing size and number of categories of data, and methods such as schema matching, instance analysis, and log mining to analyze data from a single dimension may have problems such as not making full use of the diversity of data or incomplete analysis. Addressing the above issues, this paper analyzes the data from multiple dimensions by integrating schemas, instances, and logs to make full use of the diversity of data to establish entity associations.

3 Our Customized Framework

The entity association model in Fig. 1 shows the mapping relationship between multiple sources of data from different departments in the enterprise business system and the data space. According to the multidimensional analysis framework proposed in this paper, normalized similarity values between data that can be compared are obtained to establish the association relationship between entities. As shown in Fig. 1, R1 indicates a similarity value of 1 between its associated entities a_{13} and n_{11}.

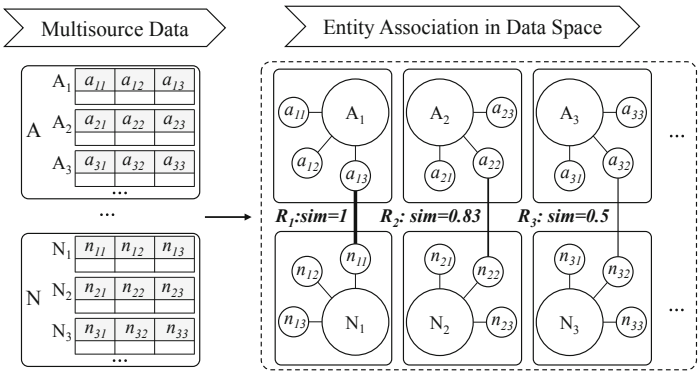


Fig. 1. Entity association mapping for multisource data

In the middle of the business logic layer and data access layer of each business system, such as ERP, CRM, and SCM, we deploy probes to obtain data. Then, the business logic layer of the data is stored as logs, and the rest of the data are stored in a relational database. To overcome the problems of large size and a variety of data types, the model pre-classifies the data based on their characteristics and nature, which improves the data processing and increases the accuracy of matching between entities. The structure and content of the data are divided into two categories, schema and instance, while logs as

a carrier of business logic are grouped into a separate category. The similarity values between the data are analyzed and calculated in three dimensions: schema, instance and log. The schema matching analysis includes both attribute names and constraints, and the instance analysis is divided into three analysis methods according to data type: numeric, character and long text. Based on the attribute association information contained in SQL, the log analysis calculates the similarity values between the data. Finally, based on the fuzzy logic analyzer, a normalization calculation is performed based on similar values for the data in different dimensions to obtain the effective association values in the data space. The corresponding schema is shown in Fig. 2.

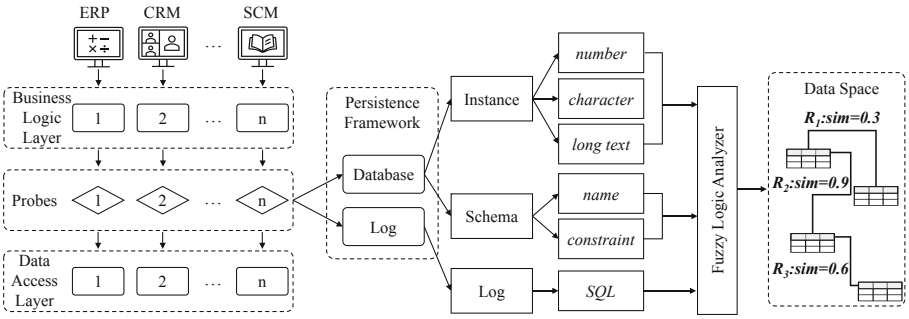


Fig. 2. A logical framework for multisource data analysis

3.1 Schema Similarity Model

Many different databases are developed by database designers to fit application scenarios, naming conventions, and other factors, but database designs generally contain table and field names, table structures, and data types. As such, the attribute names and constraints of the schema information in the database are extracted as the analysis content of the schema similarity model to measure the similarity between the data.

Name of Attribute. Attribute name analysis is divided into two types: plain text similarity and text semantic similarity analysis. The text similarity between attribute names is calculated by the edit distance algorithm, and text semantic similarity is calculated through a semantic library.

Edit distance is a way of quantifying how similar two strings are; it takes two words, w_1 and w_2, and finds the minimum number of operations required to convert w_1 to w_2. The plain text similarity value is defined according to the minimum number of edits.

$$S_{plain}(w_1, w_2) = 1 - \frac{D(w_1, w_2)}{Max(l_1, l_2)} \tag{1}$$

where l_1 and l_2 are the character lengths of w_1 and w_2, and D is the edit distance of w_1 and w_2.

Different expressions may be used for the description of the same entity. For example, if the information of an upstream company is recorded in the enterprise database, its attribute name can be named *CompanyID* and *SupplierID* based on different scenarios. To address the fact that plain text analysis cannot resolve the semantics between words, a semantic-based similarity analysis method is proposed. In particular, a tree semantic hierarchy is established for the attribute names, as shown in Fig. 3, and the similarity between words is calculated by the corresponding positions of the attribute names in the tree diagram.

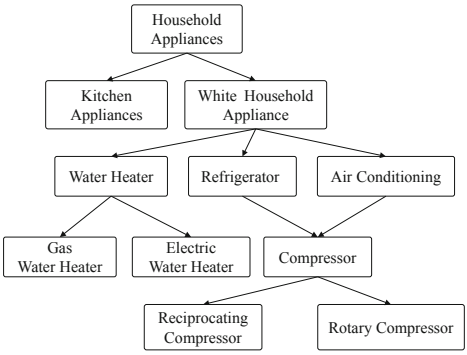


Fig. 3. Attribute name tree semantic hierarchy diagram

Therefore, the formula for calculating the semantic-based similarity is:

$$S_{sema}(w_1, w_2) = \frac{2H}{N_1 + N_2 + 2H} \tag{2}$$

where N_1 and N_2 are the shortest paths from words w_1 and w_2 to the nearest common parent word w, respectively, and H denotes the shortest path from w to the root node.

S_{name} is defined as the maximum of the plain text similarity and the semantic similarity of the text.

$$S_{name} = Max\left(S_{plain}, S_{sema}\right) \tag{3}$$

Constraint. Designers follow certain principles when programming columns in a database, such as the appropriate data type, whether it is empty, etc. The representative constraints selected from these rules can be used to explore the similarity between columns. We extracted the following constraints as features: type of each column, if the column is a primary or foreign key or not, if the column has constraint of null or not null, if the column has comments (Table 1).

Assume that the two columns requiring constraint similarity discrimination are A and B, and a_i and b_i are the values of the i-th candidate constraint corresponding to the attributes of the two columns, respectively, such that:

$$v_i = \begin{cases} 1 & a_i = b_i \\ 0 & otherwise \end{cases}, \quad i = 1, 2, ..., n \tag{4}$$

Table 1. Constraint features

$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$
Type of column	Null	Primary key	Foreign key	Comments

where n is the number of candidate constraints; then, the attribute constraint similarity between column A and column B is:

$$S_{cons} = \frac{\sum_i v_i}{n} \tag{5}$$

Schema Similarity. S_{schema} includes attribute names and constraint analysis similar values by weighting.

$$S_{schema} = \alpha \cdot S_{name} + (1 - \alpha) \cdot S_{cons} \, (\alpha \in [0, 1]) \tag{6}$$

3.2 Instance Similarity Model

Since there are similarity trends in datasets representing similar entities, such as value intervals, and keywords. It is obvious that data categories are distinctive features of a dataset. Establishing differentiated feature extraction schemes for different classes of datasets can improve the accuracy of data association matching. Generally, if the data categories are different, there is no similar relationship.

Table 2. Data type categorization

Data type	Members
exact numeric data type	SMALLINT, MEDIUMINT, INT, BIGINT
approximate numeric data type	FLOAT, DOUBLE, DECIMAL
string data types	CHAR, VARCHAR, BLOB, TEXT

According to the different data types, instance analysis can be divided into the following three types: numeric, character, and long text. The numeric type refers to the exact numeric data type and the approximate numeric data types in Table 2. The string data types are divided into two categories, character and long text, according to the length of the text. After classifying and clustering the data, the similarities between the data are analyzed according to the process shown in Fig. 4.

Number. Number is scalar; considering the similarity between columns from the perspective of numerical distribution, the mean, median, mode, standard deviation, maximum, and minimum values are selected as the feature vector elements. The feature vector corresponding to each column is calculated, substituted into the cosine similarity formula, and the result is used as the numerical similarity value.

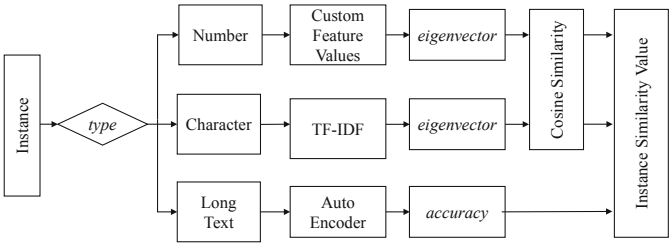


Fig. 4. Instance analyzer

Character. Character is short textual content, and it uses term frequency-inverse document frequency as the similarity calculation algorithm. First, the content of the columns that need to determine similarity is combined as a separate dataset. Then, the vectors for each column are found. Finally, the feature vectors are substituted into the cosine similarity formula to calculate the similarity value.

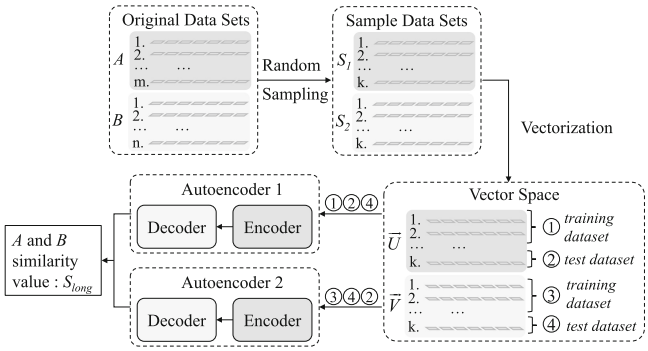


Fig. 5. The long text analysis process

Long Text. Long text is long text content, where the records in the columns are mapped as vectors, a model is built using an autoencoder, and the similarity values between columns are calculated based on the model. Assuming that A and B are two columns in the database, and they share the long text data type (Fig. 5), the overfitting problem of the model due to the large difference in the number of datasets is solved by randomly selecting k records in columns A and B as the sample data sets S_1 and S_2. Since vectors are required as input for the autoencoder, the text in the sample data sets is transformed into vectors $\overrightarrow{U}$ and $\overrightarrow{V}$. Then, the vectors are divided into a training set and test set, the autoencoder model is built using the training set, and the similarity of columns A and B is calculated according to the accuracy of the test set.

Algorithm 1. Long text similarity calculation method

$Input: x, y, \omega$
$Output: \lambda, \theta$
1: $a\_train, a\_test \leftarrow train\_test\_split(x)$
2: $b\_test \leftarrow y$
3: $a\_num, b\_num \leftarrow len(a\_test), len(b\_test)$
4: input $\leftarrow a\_train$
5: $encoded = Dense(input)$
6: $decoded = Dense(encoded)$
7: $autoencoded = Model(input, decoded)$
8: $a\_test\_predict = autoencoded(a\_test)$
9: $b\_test\_predict = autoencoded(b\_test)$
10: *For* a, b *in* $a\_test, a\_test\_predict$
11: $s\_a\_num++ \leftarrow similarity(a,b) \geq \omega$
12: $\lambda \leftarrow \frac{num}{s\_a\_num}$
13: *For* a, b *in* b$_test, b_test_predict$
14: $s\_b\_num++ \leftarrow similarity(a,b) \geq \omega$
15: $\theta \leftarrow \frac{num}{s\_b\_num}$

The autoencoder model calculates similarity, as shown in Algorithm 1. For input, x is divided into a training set and a test set according to a custom scale, y is used as the test set, and ω is the custom text similarity threshold. On output, λ and θ is the percentage of the test set evaluated as similar. For Autoencoder 1, x and y for the input in Algorithm 1 are $\overrightarrow{U}$ in vector space and the test dataset of $\overrightarrow{V}$ in vector space; the output is λ_1, θ_1. For Autoencoder 2, x and y for the input in Algorithm 1 are $\overrightarrow{V}$ in vector space and test dataset of $\overrightarrow{U}$ in vector space, the output is λ_2 and θ_2. According to the results obtained from the Autoencoder, S_{long} represents two columns of similar values:

$$S_{long} = Min\left(\frac{\theta_1}{\lambda_1}, \frac{\theta_2}{\lambda_2}, 1\right) \tag{7}$$

3.3 Log Similarity Model

The business logic layer in the layered architecture mainly packages the attributes and behaviors of entities. Although the representation of entities varies across different business logics, similar entities have similar attributes and behaviors. The SQL commands recorded in the logs contain correlation relationships between columns, which can be used as a basis of analysis for measuring column similarity. The column-to-column

similarity can be obtained by counting the number of equivalence relations in the log file.

Assuming that A and B are columns in the database, the log similarity value of columns A, B is calculated by:

$$S_{log} = \frac{N_{ab}}{N_a + N_b} \tag{8}$$

where a and b are names of columns A and B, N_a and N_b are the number of SQL commands containing a and b in the log and N_{ab} is the number of SQL commands containing both a and b in the log.

3.4 Fuzzy Logic Similarity

The similarity values obtained from the above calculation by schema, instance, and log similarity models are processed using fuzzy logic for standardization.

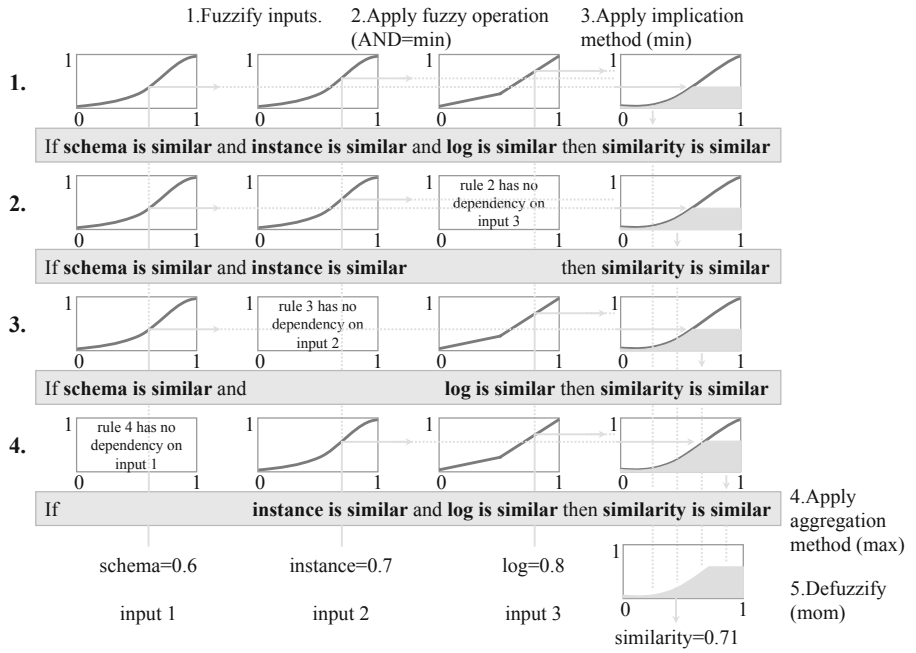


Fig. 6. Fuzzy logic instance diagram

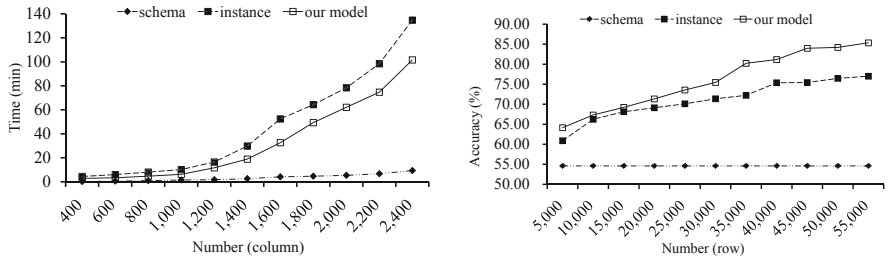
While A and B are the columns in the database, the similarity values obtained from the above three-dimensional analysis are substituted into the affiliation function to obtain the affiliation values. The values that meet the fuzzy rules are aggregated according to the rules and defuzzified to obtain a normalized measure of column-to-column similarity. In Fig. 6, for example, A and B have similar values of 0.6, 0.7, and 0.8 in the schema, instance, and log dimensions. Through a series of fuzzy operations, the similarity value between A and B is 0.71.

4 Experiment

To verify the feasibility of the proposed framework, this paper uses data from all business systems of a company and stores them in a unified manner. The hardware environment for the experiments is an Intel(R) Xeon(R) Silver 4210 CPU @ 2.20 GHz, 64 GB RAM, RTX2080Ti*4. The results are the average of three replicated experiments. The dataset consists of Haier and public data set available on the Internet. It mainly includes the following categories: product data, enterprise operation data, value chain data, and external data [12].

4.1 Comparison of Experiments for the Different Solutions of Data Space Entity Association

A certain number of columns are randomly selected as samples from all data, the number of rows in each column is different, and experiments are conducted using schema matching (see Subsect. 3.1), instance analysis (see Subsect. 3.2), and the fuzzy logic-based model proposed in this paper to compare them in two ways: running time and accuracy.



(a) Running time of different methods (b) Accuracy of different methods

Fig. 7. Comparison of experiments for different methods

Experiments based on the schema take less time, as seen in Fig. 7(a). The instance-based method takes significantly more time for the same amount of data due to the comprehensive content analysis, while the method proposed in this paper includes instance analysis but takes less time than the instance-based method because the data are analyzed in categories during the instance analysis.

The accuracy of the schema-based method is found to remain unchanged when the volume of data is changed, as seen in Fig. 7(b), because the elements analyzed are constant; with an increase in the volume of data, the accuracy of the left two analysis methods generally tends to increase, but the method proposed in this paper maintains the highest accuracy in each experiment since data is preclassified.

As shown in Fig. 7, the proposed method in this paper can obtain a high accuracy rate in a short time with a moderate amount of data.

5 Conclusion

This paper proposes a hybrid data matching model based on schema, instances, and logs. The model consists of four main components: the front probe to acquire the analysis data, the analysis data, the three-dimensional outputs, and the normalized metric based on fuzzy logic. Experimental results show that the model provided in this paper has better results in terms of accuracy and efficient handling of mass data compared to previous single matching methods based on schema or instances. For further research, the focus is on how to establish a mapping relationship between data and weights and on establishing a guidance scheme for weight assignment to better address the impact of the randomness of multisource heterogeneous data on the accuracy of the results.

Acknowledgement. This work was supported in part by the National Key R&D Program of China under Grant 2018YFB1702902, and in part by the Shandong Province Colleges and Universities Young Talents Initiation Program under Grant 2019KJN047.

References

1. Li, G.: Research on innovation of enterprise management accounting informatization platform based on intelligent finance. In: Proceedings of the 1st International Symposium on Economic Development and Management Innovation (EDMI 2019), pp. 286–291. Atlantis Press, Paris (2019). https://doi.org/10.2991/edmi-19.2019.47
2. Nakamura, E.F., Loureiro, A.A.F., Frery, A.C.: Information fusion for wireless sensor networks: methods, models, and classifications. ACM Comput. Surv. **39** (2007). https://doi.org/10.1145/1267070.1267073
3. Boström, H., et al.: On the definition of information fusion as a field of research. IKI Tech. Reports. 1–8 (2007)
4. Alwan, A., Nordin, A., Alzeber, M., Zaid, A.: A survey of schema matching research using database schemas and instances. Int. J. Adv. Comput. Sci. Appl. 8 (2017). https://doi.org/10.14569/ijacsa.2017.081014
5. Li, X., Wen, H., Hu, Y., Jiang, L.: A novel beta parameter based fuzzy-logic controller for photovoltaic MPPT application. Renew. Energy **130**, 416–427 (2019). https://doi.org/10.1016/j.renene.2018.06.071
6. Roumila, Z., Rekioua, D., Rekioua, T.: Energy management based fuzzy logic controller of hybrid system wind/photovoltaic/diesel with storage battery. Int. J. Hydrogen Energy **42**, 19525–19535 (2017). https://doi.org/10.1016/j.ijhydene.2017.06.006
7. Tan, W., Mapforce, A.: Approximation Algorithms for Schema-Mapping Discovery. 42 (2017)
8. Wu, J., Pan, S., Zhu, X., Zhang, C., Wu, X.: Multi-instance learning with discriminative bag mapping. IEEE Trans. Knowl. Data Eng. **30**, 1065–1080 (2018). https://doi.org/10.1109/TKDE.2017.2788430
9. Gomes Dos Reis, D., Ladeira, M., Holanda, M., De Carvalho Victorino, M.: Large database schema matching using data mining techniques. In: IEEE International Conference on Data Mining Workshops, ICDMW, 2018-November, pp. 523–530 (2019). https://doi.org/10.1109/ICDMW.2018.00083
10. Bakhtouchi, A.: Data reconciliation and fusion methods: a survey. Appl. Comput. Informatics. 1–7 (2019). https://doi.org/10.1016/j.aci.2019.07.001

11. Mohanty, S.N., Rejina Parvin, J., Vinoth Kumar, K., Ramya, K.C., Sheeba Rani, S., Lakshmanaprabu, S.K.: Optimal rough fuzzy clustering for user profile ontology based web page recommendation analysis. J. Intell. Fuzzy Syst. **37**, 205–216 (2019). https://doi.org/10.3233/JIFS-179078
12. https://github.com/forgstfree/Industrial-Datasets. Accessed 06 May 2021