Multi-Pattern Matching Method Based on SimHash and Hybrid Similarity (Postprint)

Authors: Cao Weidong, Hu Wei, Wang Jialiang, Wang Jing

Date: 2018-11-29T00:00:00+00:00

Abstract

To address the problems of low efficiency, insufficient accuracy, and difficulty in obtaining complete schema information in multi-schema matching during the integration of multi-source heterogeneous civil aviation passenger service data, this paper proposes a multi-schema matching method based on SimHash and hybrid similarity. The method first calculates feature unit weights based on PMI and constructs signatures for attribute columns through the SimHash algorithm to represent attribute features, thereby reducing feature dimensionality. It then introduces the K-means++ algorithm to cluster attributes and generate candidate matching sets. Finally, an attribute mapping graph is constructed based on the hybrid similarity of attributes to intuitively display the matching relationships between attributes while improving the efficiency of multi-schema matching. Experimental results demonstrate the feasibility of the method, providing a new solution for efficiently resolving schema conflict issues in the integration of multi-source heterogeneous civil aviation passenger service data.

Full Text

Preamble

Multiple Schema Matching Method Based on SimHash and Mixed Similarity

Cao Weidong, Hu Wei, Wang Jialiang, Wang Jing
(College of Computer Science & Technology, Civil Aviation University of China, Tianjin 300300, China)

Abstract: To address the challenges of low efficiency, insufficient accuracy, and difficulty in obtaining complete schema information during multi-source heterogeneous civil aviation passenger service data integration, this paper proposes a multiple schema matching method based on SimHash and mixed similarity. The method first calculates feature unit weights using Pointwise Mutual Information

(PMI) and constructs attribute column signatures via the SimHash algorithm to represent attribute features, thereby reducing feature dimensionality. It then employs the K-means++ algorithm to cluster attributes and generate candidate matching sets. Finally, it constructs an attribute mapping graph based on mixed similarity to visually display matching relationships between attributes while improving multi-schema matching efficiency. Experimental results demonstrate the feasibility of the proposed method, providing a novel solution for efficiently resolving schema conflicts in multi-source heterogeneous civil aviation passenger service data integration.

Keywords: multiple schema matching; signature; PMI; mixed similarity; attribute mapping graph

Classification: TP391
DOI: 10.19734/j.issn.1001-3695.2018.06.0462

0 Introduction

Traditional schema matching methods face several problems when applied to civil aviation passenger service data integration. First, schema information-based matching methods require complete schema information. Domestic civil aviation passenger service information systems contain substantial revenue-related data such as PNR (Passenger Name Record), ET (Electronic Ticket), and CKI (Check-In information). When schema information is incomplete, relying solely on textual similarity between attributes yields unsatisfactory matching results. For example, the attributes `psg_{type}` in Table 1 and `opt_{type}` in Table 2 exhibit high textual similarity but represent completely different meanings. Second, instance-based matching methods cannot resolve false matching caused by similar data distribution characteristics. For instance, `orgn_{city}` in Table 1 represents departure city while `destination` in Table 2 represents arrival city; their data instance distributions are similar, but judging them as semantically identical would lead to errors. Third, traditional methods address binary matching problems. When matching n schemas, conventional approaches match two schemas at a time, requiring $n(n-1)/2$ iterations, resulting in low efficiency.

Due to high data security requirements in civil aviation passenger services, multi-level security access permissions make detailed schema information difficult to obtain. Under these constraints where complete schema information is inaccessible, the aforementioned four categories of methods struggle to achieve satisfactory matching results. Therefore, this paper proposes a method that uses data instances and attribute names as auxiliary matching information, which remains applicable in security-sensitive civil aviation domains.

1 Related Work

Schema matching methods have achieved considerable success over years of development [1]. Based on different auxiliary matching information, they primarily fall into four categories:

a) Schema information-based matching: Methods such as COMA [2] and SEMINT [3] combine multiple schema information sources to achieve good matching results, but suffer from high time complexity when matching multiple schemas. Ding et al. [4] proposed constructing attribute feature vectors using TF-IDF for clustering analysis, which reduces time complexity in multi-schema matching but performs poorly due to "same name, different meaning" and "different name, same meaning" attributes.

b) Structure information-based matching: Early approaches represented source and target schemas as tree or graph structures, then computed similarity between corresponding nodes to select optimal matches [5-6]. To improve accuracy, Du et al. [7] later proposed the IU_{Based} method.

c) Instance-based matching: Early instance-based methods discovered matching relationships by analyzing data instance repetition patterns [8, 9], but the distribution characteristics they mined were incomplete. Ahmadi et al. [10] proposed using q-gram combined with mutual information theory to construct attribute column feature vectors, but this leads to false matches when data instances are similar (e.g., between numeric attribute columns). Mehdi et al. [11] proposed using regular expressions to match data instances, but their method only employed Google similarity to distinguish false matches, limiting accuracy. Gu et al. [12] proposed interactively executing instance matching and schema matching to improve accuracy, though this complicates the matching process.

d) Methods using other auxiliary information: Approaches utilizing ontology knowledge to construct schema ontologies for matching with global ontologies [13] provide new ideas for multi-schema matching, but require multiple schema information sources that may not be accessible under data access restrictions.

2.1 PMI-Based SimHash Algorithm

In multi-schema contexts, attribute data instances are referred to as attribute columns. Due to the one-to-one correspondence between attribute columns and attributes, attribute column features can represent attribute features. Traditional methods using mutual information theory to compute attribute similarity suffer from high-dimensional feature vectors due to numerous extracted feature units, hindering subsequent computation. This paper employs a PMI-based SimHash algorithm to generate fixed-length signatures as attribute column features, effectively reducing feature vector dimensionality. Related definitions are provided below.

Definition 1 (Feature Unit): A numerical value or string extracted from data instances that carries actual meaning and can represent data instance characteristics. Since data instances are complex and variable, they are categorized into string, temporal, and numeric types for feature unit extraction. For string-type data, q-gram extracts feature units according to Definition 1. Temporal data is processed by splitting into year, month, day, hour, minute, second units. Numeric data, being sparse, uses equal-interval partitioning for feature unit extraction. After extracting feature units from attribute column a, they are represented as key-value pairs $a = \{(u_1,\ t_1),\ (u_2,\ t_2),\ \cdots,\ (u\ ,\ t\ )\}$, where u is a feature unit of a and t is the frequency of u in attribute column a. The intersection of all feature units from n attribute columns forms the feature set $U = \{u_1,\ u_2,\ \cdots,\ u\ \}$.

Definition 2 (Pointwise Mutual Information): A measure of the information content difference between any attribute column a and any feature unit u, denoted as $pmi(a\ ,\ u\ )$:

$$pmi(a_k, u_y) = \log\left(\frac{p(a_k, u_y)}{p(a_k)p(u_y)}\right) = \log\left(\frac{\frac{t_{ky}}{\sum_{i=1}^{n} t_{iy}} \cdot \frac{\sum_{i=1}^{n} t_{iy}}{\sum_{k=1}^{n}\sum_{y=1}^{m} t_{ky}}}{\frac{\sum_{y=1}^{m} t_{ky}}{\sum_{k=1}^{n}\sum_{y=1}^{m} t_{ky}} \cdot \frac{\sum_{i=1}^{n} t_{iy}}{\sum_{k=1}^{n}\sum_{y=1}^{m} t_{ky}}}\right)$$

where t represents the frequency of feature unit u in attribute column a, $_1{}^n$ t represents the total frequency of feature unit u across all attribute columns, $_1$ t represents the total frequency of all feature units in attribute column a, and T represents the total frequency of all feature units across all attribute columns.

The SimHash algorithm [14] is an efficient method for computing similarity among massive texts, converting high-dimensional text features into fixed-length signatures. From Definition 2, larger PMI values between an attribute column and its feature units indicate stronger correlation. If two attribute columns share more identical feature units, they are more likely to match. Therefore, this paper uses PMI values between feature units and attribute columns as weights, proposing a PMI-SimHash attribute column signature generation algorithm.

Algorithm 1: Generate Attribute Column Signatures

Input: Set of n attribute columns $A = \{a_1,\ a_2,\ \cdots,\ a\ \}$.

Output: Signature set P for all attribute columns.

1. Initialize $P =$
2. For each a A:
   - Extract feature unit set $a = \{u_1,\ u_2,\ \cdots,\ u\ \}$
   - For each u a:
     - Generate f-bit signature $s = hash(u)$
     - For each bit i in s:
       - If s $==$ *1*: s $= pmi(a,\ u)$

        * Else: s = *-pmi(a, u)*
    - Sum all signatures s for feature units in a bitwise to obtain S
    - For each bit i in S:
        - If $S > 0$: $S = 1$
        - Else: $S = 0$
    - $P = P.add(S)$
3. Return P

The algorithm iterates through attribute column set A, extracts feature unit set $a = \{u_1, u_2, \cdots, u\}$, generates f-bit signature s for each feature unit u using a hash function, and computes *pmi(a, u)* using Equation (1). If the i-th bit of s is 1, it updates the i-th bit to *pmi(a, u)*; otherwise, it updates it to *-pmi(a, u)*. After processing all feature units in a, it performs bitwise summation to obtain S, updates S values, adds the attribute column signature S to set P, and finally returns P.

2.2 Clustering Analysis

Due to the one-to-one correspondence between attributes and attribute columns, clustering analysis using attribute column signatures reveals attribute clustering relationships. The K-means++ algorithm [15], a partition-based clustering method, offers fast convergence and high stability compared to standard K-means. This paper uses signature set P and cluster count k as inputs to K-means++, outputting set $R = \{r_1, r_2, \cdots, r\}$ containing k candidate matching sets, where r is the i-th candidate matching attribute set.

As k varies, two clustering outcomes may occur: (a) attributes with different semantics may cluster together, or (b) attributes with identical semantics may belong to different clusters. For problem (a), this paper proposes an attribute mixed similarity calculation method to distinguish semantically inconsistent attributes within candidate matching sets. For problem (b), it employs a heuristic approach to dynamically optimize the k value.

3 Multi-Schema Matching Based on Mixed Similarity

To more accurately distinguish false matches in candidate matching sets, this section proposes a novel mixed similarity calculation model based on similarity distinguishability capability, then constructs an attribute mapping graph to compute attribute mixed similarity and describe matching relationships.

3.1.1 Syntax and Semantic Attribute Similarity Calculation

Civil aviation passenger service data attributes often appear in stem or compound word forms. After computing syntactic similarity using TF-IDF, standardization through splitting and lemmatization is required before semantic similarity calculation (e.g., *TICKET_{NO}* → {ticket, number}).

a) TF-IDF-based syntactic similarity calculation [4]: Attributes in candidate matching sets are first segmented into letter units using q-gram, then TF-IDF computes letter unit weights w, representing attributes as weight vectors $v = (w_1, w_2, \cdots, w)$. Syntactic similarity between attributes sn and sn is:

$$EdSim(sn_i, sn_j) = \frac{v_i \cdot v_j}{\|v_i\| \times \|v_j\|}$$

b) WordNet-based semantic similarity calculation [16]: In WordNet, two factors affect concept word similarity: distance between concepts in WordNet and information content (IC) of concept words. IC is influenced by concept depth (positively correlated) and density (negatively correlated) in WordNet. This paper uses an IC-based similarity model:

$$IC(sn) = \log\left(\frac{\text{depth}(sn)^{\lambda} \times e^{-\text{depth}(sn)} + e^{-\lambda \times \text{hypo}(sn)}}{\text{Node}_{\max}}\right)$$

$$L(IC)_{i,j} = IC(sn_i) + IC(sn_j) - 2 \times IC(\text{LCS}(sn_i, sn_j))$$

$$L(\text{path})_{i,j} = \frac{\log(\text{Dis}(sn_i, sn_j) + 1)}{\log(2 \times \text{Depth}_{\max})}$$

$$WtSim(sn_i, sn_j) = e^{-\alpha \times L(IC)_{i,j} + \beta \times L(\text{path})_{i,j}}$$

where *IC(sn)* represents information content of attribute *sn*, *L(IC)* represents IC semantic distance, *L(path)* represents shortest path-based semantic distance, *hypo(sn)* represents the number of hyponyms for *sn* in WordNet, *Node* represents total concept nodes in WordNet, *depth(sn)* represents depth of *sn* in WordNet, *Dis(sn , sn)* represents shortest distance between *sn* and *sn* in WordNet, and λ, α, β are positive parameters.

For non-compound attributes, semantic similarity is computed directly using Equation (6). For compound attributes, after decomposition into word sets containing two or more words, word set similarity is computed using Equation (6), where *WtSim(sn , sn)* represents the word set similarity.

3.1.2 Mixed Similarity Calculation Model

Using syntax or semantic similarity alone cannot accurately represent attribute relationships. Therefore, this paper proposes a new mixed similarity model. For a labeled matching attribute pair, if syntactic (or semantic) similarity approaches 1, that similarity measure has strong distinguishability. For a labeled

non-matching pair, if similarity approaches 0, that measure also demonstrates strong distinguishability. Based on this analysis:

Definition 3 (Similarity Distinguishability Capability): For labeled attribute pair sets X (all matching) and X (all non-matching), let *SIM_X* and *SIM_X* represent similarity sets from syntactic (or semantic) methods. Distinguishability capability is:

$$dpsim = \frac{\sum_{sim_i \in SIM_{X_m}} sim_i - \sum_{sim_i \in SIM_{X_u}} sim_i}{|X_m| + |X_u|}$$

where *dpsim* represents distinguishability capability, *sim* represents attribute pair similarity, and $|X|$ and $|X|$ represent set sizes.

The mixed similarity model based on distinguishability capability is:

$$sim(sn_i, sn_j) = \left(\frac{dpsim(EdSim)}{dpsim(EdSim) + dpsim(WtSim)} \times EdSim(sn_i, sn_j)^p + \frac{dpsim(WtSim)}{dpsim(EdSim) + dpsim(WtSim)} \right.$$

where *dpsim(EdSim)* and *dpsim(WtSim)* represent syntactic and semantic distinguishability capabilities, and p is a positive parameter.

3.2 Multi-Schema Matching Algorithm Based on Mixed Similarity

To filter false matches from candidate matching sets and obtain final attribute matching relationships, this section constructs an attribute mapping graph based on mixed similarity (Algorithm 2).

Algorithm 2: Construct Attribute Mapping Graph G(R, E)

Input: Matching attribute pair set X, non-matching set X, thresholds and , candidate matching set $R = \{r_1, r_2, \cdots, r\}$.

Output: Attribute mapping graph *G(R, E)*.

1. Initialize *dpsim(EdSim)'* = *dpsim(WtSim)'* = *0.5*, $M =$, $U =$
2. For *xm* X and *xu* X:
   - $M = M.add(xm)$, $U = U.add(xu)$
   - If |*dpsim(EdSim)* - *dpsim(EdSim)'* | < : return *dpsim(EdSim)*
   - Else: *dpsim(EdSim)'* = *dpsim(EdSim)*
   - If |*dpsim(WtSim)* - *dpsim(WtSim)'* | < : return *dpsim(WtSim)*
   - Else: *dpsim(WtSim)'* = *dpsim(WtSim)*
3. Initialize edge set $E =$
4. For each r R:
   - For *sn* , *sn* r:
     - Compute *EdSim(sn , sn)* and *WtSim(sn , sn)*
     - Compute mixed similarity *sim(sn , sn)*

– If *sim(sn , sn) ≥ : E = E.add((sn , sn))*

5. Return *G(R, E)*

The algorithm first randomly selects attribute pairs from labeled matching and non-matching sets to compute TF-IDF and WordNet similarities iteratively until distinguishability changes fall below , returning *dpsim(EdSim)* and *dpsim(WtSim)*. It then traverses R, computes similarities for attribute pairs, and adds edges for pairs exceeding threshold , outputting graph *G(R, E)* where matching attributes are connected and false matches remain isolated points.

Complexity analysis reveals that COMA' s time complexity is $O(n^2m^2)$ for n schemas with m attributes each, as it requires $Y_1 = n(n-1)/2 \times m^2$ similarity computations. Our method first clusters $m \times n$ attributes into k candidate sets, then computes similarities within each set. With $Y_2 = \Sigma_1 (x(x-1)/2)$ and typical $k \quad n$, $x \quad m$, we obtain $Y_2 \quad n(m^2 - m)/2$. Since $Y_1 > Y_2$, our algorithm reduces computation. The time complexity is $O(nm^2)$, lower than COMA' s $O(n^2m^2)$.

4 Experiments

4.1 Experimental Dataset

Data from four civil aviation passenger service system sources were used: Passenger Name Records (PNR), Electronic Tickets (ET), Check-In (CKI), and Inventory (INV) data. These sources have different schemas, with various functional modules containing redundant attributes that differ in name but share semantics. Table 3 shows the dataset composition.

Table 3: Number of Attributes and Instances

Heterogeneous Data Source	Attribute Count	Instances per Attribute	Matching Attribute Count
PNR	16	30,000	10
ET	14	30,000	10
CKI	15	30,000	10
INV	12	30,000	10

4.2 Evaluation Metrics

Performance is evaluated using precision, recall, and overall metrics. Let T be correct matches returned, P be all matches returned, F be false matches, and R be actual correct matches:

$$\text{Precision} = \frac{T}{P}, \quad \text{Recall} = \frac{T}{R}, \quad \text{Overall} = \frac{T}{P} \times \frac{T}{R}$$

4.3.1 Impact of k Value on Clustering Results

Cluster count k significantly affects results. Too small a k reduces precision by merging dissimilar classes; too large a k reduces recall by splitting similar attributes. With 500 instances and K-means++, results show that when $k = 15$, the number of perfectly matched attributes peaks, indicating optimal clustering. Since the maximum attributes in a single schema is 16 (Table 3), k should approximate the maximum attribute count per schema.

4.3.2 Distinguishability of Different Similarity Measures

Equal-sized matching and non-matching attribute pair sets were constructed. Using WordNet parameters $\lambda = 0.4$, $\alpha = 0.2$, $\beta = 0.1$ from literature [16], results show WordNet's distinguishability stabilizes around 0.69, while TF-IDF's increases then stabilizes around 0.61. TF-IDF is less sensitive to synonym-based attributes, making it weaker than WordNet.

4.3.4 Impact of Instance Quantity on Matching Results

Experiments with varying instance quantities (500 to 30,000) show overall metrics rising weakly. With 500 instances, overall = 0.793; with 1,000 instances, overall = 0.751 (due to uneven distribution); with 30,000 instances, overall reaches 0.830. More instances yield higher accuracy.

4.3.5 Comparative Experimental Analysis

Our method (B_{SHM}) is compared against: (1) B_{ATT} [4] using TF-IDF features, and (2) B_{INS} [10] using q-gram and mutual information vectors.

Experiment 1: With $k = 15$, $= 0.45$, *30,000 instances*, Table 4 shows B_{SHM} achieves highest precision (0.951), matching recall (0.875) with B_{INS}, and highest overall (0.830)—improving 0.292 over B_{ATT} and 0.156 over B_{INS}. B_{ATT}' s reliance on syntactic similarity suffers from homonym/polysemy issues, while B_{INS}' s instance-only approach fails when different semantics share similar instance features. B_{SHM} overcomes both limitations.

Table 4: Comparison of Three Methods

Method	Precision	Recall	Overall
B_{SHM}	0.951	0.875	0.830
B_{ATT}	0.758	0.791	0.538
B_{INS}	0.813	0.875	0.674

Experiment 2: Runtime tests with 2, 3, and 4 schemas (Table 5) show B_{INS} is slowest, B_{SHM} moderate, and B_{ATT} fastest. B_{SHM}' s additional similarity computation after clustering increases time over

B_{ATT}, but its 128-bit fingerprint conversion makes clustering much faster than B_{INS}' s high-dimensional vectors. Overall, B_{SHM} offers the best performance for civil aviation passenger service data integration.

Table 5: Runtime Comparison

Schema Count	B_{SHM}	B_{ATT}	B_{INS}
2	0.303s	0.135s	0.576s
3	0.412s	0.216s	1.296s
4	0.547s	0.324s	2.304s

5 Conclusion

Analyzing multi-source heterogeneous civil aviation passenger service data, this paper proposes a multi-schema matching method based on SimHash and mixed similarity to address low efficiency and accuracy issues. The method uses attribute column signature clustering to generate candidate matching sets, avoiding "same name, different meaning"and "different name, same meaning"problems. A more accurate mixed similarity model balances errors from single-similarity measures. Finally, attribute mapping relationships based on mixed similarity effectively distinguish false matches caused by similar instance features while avoiding cumbersome complete schema information acquisition. Experiments demonstrate that the proposed method effectively resolves multi-schema matching inefficiency and inaccuracy in multi-source heterogeneous data integration, offering significant value for civil aviation passenger service data integration.

References

[1] Zheng Wenyi, Ju Shiguang. Research on schema matching approaches [J]. Application Research of Computers, 2006, 23(2): 60-63.

[2] Do Honghai, Rahm E. COMA: a system for flexible combination of schema matching approach [C]// Proc of the 28th International Conference on Very Large Data Bases. San Francisco, CA: Morgan Kaufmann Publishers Inc, 2002: 610-621.

[3] Li WenSyan, Clifton, et al. SEMINT: a tool for identifying attribute correspondences in heterogeneous databases using neural networks [J]. Data & Knowledge Engineering, 2000, 33(1): 49-84.

[4] Ding G, Sun T, Xu Y. Multi-schema matching based on clustering techniques [C]// Proc of the 10th International Conference on Fuzzy Systems and Knowledge Discovery. Piscataway, NJ: IEEE Press, 2013: 778-782.

[5] Melnik S, Garcia Molina H, Rahm E. Similarity flooding: a versatile graph matching algorithm and its application to Schema matching [C]// Proc of the

18th International Conference on Data Engineering. Washington, DC: IEEE Computer Society, 2002: 117-128.

[6] Madhavan J, Bernstein P A, Rahm E. Generic schema matching with cupid [C]// Proc of the 27th International Conference on Very Large Data Bases. San Francisco, CA: Morgan Kaufmann Publishers Inc. 2001: 49-58.

[7] Du Xiaokun, Li Guohui, Wang Jiangqing, et al. Schema matching method based on information unit [J]. Journal of Software, 2015, 26(10): 2596-2613.

[8] Bilke A, Naumann F. Schema matching using duplicates [C]// Proc of the 21th International Conference on Data Engineering. Washington, DC: IEEE Computer Society, 2005: 69-80.

[9] Dhamankar R, Lee Y, Doan A, et al. IMAP: discovering complex semantic matches between database schemas [C]// Proc of ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 2004: 383-394.

[10] Ahmadi B, Hadjieleftheriou M, Seidl T, et al. Type-based categorization of relational attributes [C]// Proc of the 12th International Conference on Extending Database Technology: Advances in Database Technology. New York: ACM Press, 2009: 84-95.

[11] Mehdi O, Ibrahim H, Affendey L. An approach for instance based schema matching with google similarity and regular expression [J]. International Arab Journal of Information Technology, 2017, 14(5): 755-763.

[12] Gu B, Li Z, Zhang X, et al. The interaction between schema matching and record matching in data integration [J]. IEEE Trans on Knowledge & Data Engineering, 2016, 29(1): 186-199.

[13] Shi Haohong, Yang Weidong. Research on method of multiple data sources schema matching based on global ontology [J]. Journal of Chinese Computer Systems, 2016, 37(6): 1148-1152.

[14] Manku G S, Jain A, Sarma A D. Detecting near-duplicates for web crawling [C]// Proc of the 16th International Conference on World Wide Web. New York: ACM Press, 2007: 141-150.

[15] Arthur D, Vassilvitskii S. K-means++: the advantages of careful seeding [C]// Proc of the 18th ACM-SIAM Symposium on Discrete Algorithms. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2007: 1027-1035.

[16] Zhang Siqi, Xing Weiwei, Cai Yuanyuan. A WordNet-based hybrid semantic similarity measurement [J]. Computer Engineering and Science, 2017, 39(5): 971-977.

Note: Figure translations are in progress. See original paper for figures.