ISSN 2277-3061

Feedback Based Conflict Identification and Resolution using Duplicate Elimination and Ranking Techniques

I.CAROL , Dr.S.BRITTO RAMESH KUMAR
Research Scholar, Department of computer science, St.Joseph'sCollege, Trichy
carolcrl23@gmail.com
Assistant Professor, Department of computer science, St.Joseph'sCollege, Trichy

ABSTRACT

Increase in the amount of data provides a huge scope for data analysts to operate and leverage information from them. Problems arise when the data varies in formats and their storage mechanisms become heterogeneous. Hence integration of data and its conversion to a common structure becomes mandatory. This paper presents a mechanism that operates on heterogeneous data sources, identifies conflicts and resolves them using duplicate elimination and ranking techniques. Further, a feedback mechanism is incorporated into the architecture, using which reinforcement learning is imposed on the architecture. This makes the proposed framework a machine learning architecture that is flexible and adapts according to the dynamic environment, which has become a de facto in the current scenario.

Indexing terms/Keywords

Conflict Resolution; Duplicate elimination; Ranking; Reinforcement Learning

Council for Innovative Research
Peer Review Research Publishing System
Journal: INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY
Vol. 14, No. 9
www.ijctonline.com, editorijctonline@gmail.com

INTRODUCTION

Information age has led to a huge increase in the amount of data being generated. Lowered cost of memory devices and increase in the read/write capacity of these devices has led to an increase in storage of the generated data. Complexities arise in terms of processing the data, when these data sources contain same data. It also becomes mandatory for the application to utilize all the available data before concluding a result. Hence the process of data integration becomes mandatory.

Data integration process has three major goals as to increase the correctness, completeness and to make it concise [10]. Correctness is measured in terms of whether the data confirms to the real world standard. Completeness is measures in terms of the data present in the records. Conciseness measures the uniqueness of the data. While achieving correctness and conciseness are non-trivial, achieving completeness can be achieved by using multiple data sources.

Conflicts are the inconsistencies and irrelevancies in data from various data sources corresponding to a single entity. Data conflicts are of two categories; caused due to uncertainty and the conflicts caused due to contradiction. The major questions that arise in terms of solving these conflicts are how to find the best value among the conflicting values? How to find it efficiently? Conflicts occur due to missing data and contradictions and can be resolved using any of the following methods mentioned in Figure 1.

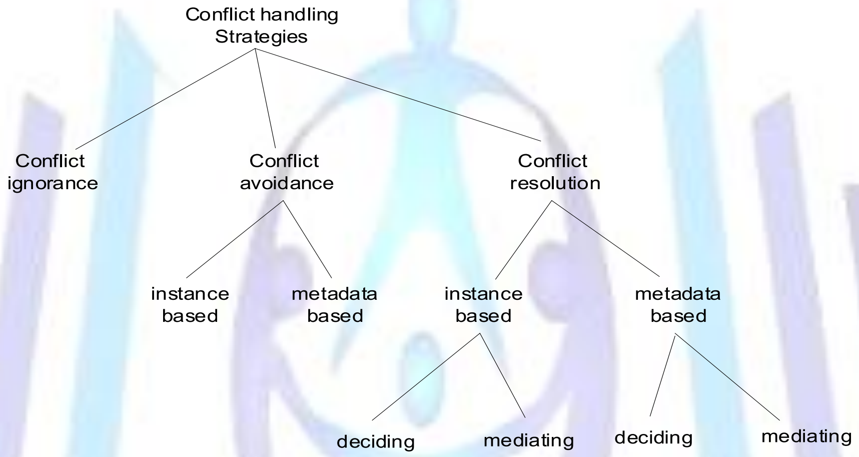


Fig 1: A Classification of the Conflict Resolution Strategies

A measure to identify the contribution of individual judgments towards inconsistency in pairwise comparisons is proposed in [11]. It addresses the need to provide appropriate measures and proposes two measures, namely congruence and dissonance. Congruence helps find the contribution of individual judgments towards the overall inconsistency in a PCM, while dissonance supplements congruence and detects outliers and the phenomena of consistency deadlock in the system. A similar inconsistency measuring technique is presented in [12] that considers a huge streaming data. It considers a stream based inconsistency measurement technique that presents a novel inconsistency measure that can be applied to streaming data, and stream based approximation techniques. This paper presents techniques that effectively identifies conflicts, detects and eliminates duplicates and finally ranks the results for precision. It also incorporates a feedback mechanism that enables reinforcement learning for providing accuracy improvements. A layered scheme for identifying inconsistencies for context aware systems is presented in [13]. A similar method considering global conflicts is presented in [14]. A semantic based inconsistency resolution system that uses automatic ontology merging system is presented in [15].

The remainder of this paper is structured as follows; section II presents the system architecture, section III provides a detailed structure of the feedback based inconsistency resolution technique, section IV presents the results and section V concludes the study.

SYSTEM ARCHITECTURE

Feedback based conflict identification and resolution using duplicate elimination and ranking techniques is a method that is based on the Reinforcement learning technique that uses the feedback mechanism from the user to identify their preferences and filter out only the necessary results. The initial phase deals with designing appropriate wrappers to query appropriate data sources. The result sets are then shortlisted and the final pruned results are integrated into the data dictionary for further querying. The results are finally ranked using the data properties. User feedback is obtained and appropriate weight alteration mechanisms are carried out to reinforce the user's preferences. Figure 2 presents the system architecture of the conflict identification and resolution technique.

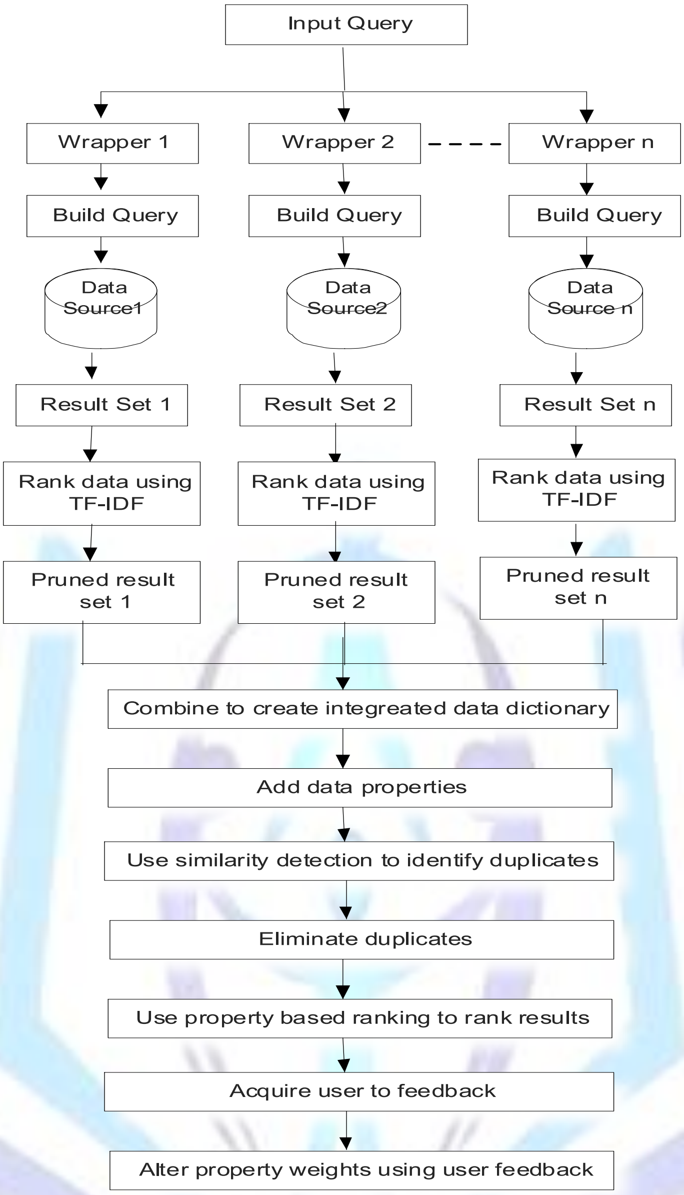


Fig 2: Conflict Identification and Resolution: Architecture

FEEDBACK BASED CONFLICT IDENTIFICATION AND RESOLUTION USING DUPLICATE ELIMINATION AND RANKING TECHNIQUES

The feedback based conflict identification and detection using duplicate elimination and ranking techniques uses enhanced ranking and similarity identification techniques to provide appropriate ranked results. The query presented by the user is either in terms of a single keyword or keyword phrase. This query serves as the major retrieval term. The query is passed to the initial phase of the processing architecture of the information retrieval system.

Exclusive query building using wrappers

The input query should be modified and should be made compatible to the data source being queried upon, in order to build an appropriate working query. Every data source is distinct in its own ways, hence the queries are also different, while a database requires query in the format of the SQL, an XML dataset requires tag processing queries, while JSON documents require queries specific to their structure. This complicates the process of integrating information from a multitude of data sources. Wrappers are designed to perform this operation for the user automatically. Wrappers are designed according to the data source that is present in its next level. Wrappers convert the data and prepare the query Z `Q12345/7 language used to process the data. The prepared query is passed to the data source and the results obtained are passed to the next level for pruning. Multiple wrappers are employed in the conflict detection architecture depending on the data sources being used. The algorithm below presents the working of wrappers.

Algorithm:

1. *Input query phrase from the user*
2. *Query phrase analysis*
3. *Stopword pruning to remove abstract components*
4. *Reform query phrase to the format accepted by the base data source*
5. *Query construction in accordance with the base data source*
6. *Ambiguity analysis in the construction of the query*
7. *Query execution in the data source*
8. *Result preparation for the next phase*

Data pruning

Though appropriate data are retrieved by the wrappers (depending on the accuracy of the query language being involved), the data still tends to contain unnecessary details, or data with less promising information. Such information is filtered in this phase. Data pruning phase uses the concept of TFIDF (Term Frequency-Inverted Document Frequency) to rank the results sets for pruning unnecessary data.

Term Frequency (TF) and the Inverted Document Frequency (IDF) are calculated using (1) and (2) [2,3]. The TF and IDF are numerical statistics that are used to identify the importance of a word in relation to a corpus or a text repository.

$$tf(t,d) = \frac{f(t,d)}{count(w,d)} \qquad (1)$$

where *f(t,d)* refers to the number of times the word *t is present* in the document *d* and *count(w,d)* refers to the number of words in the document *d*.

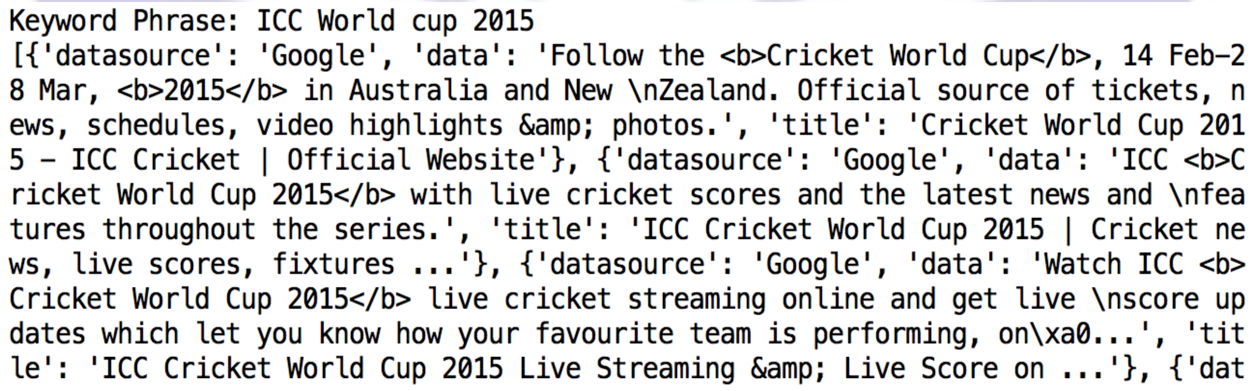
$$idf(t,D) = log\frac{N}{|\{d\in D: t\in d\}|} \qquad (2)$$

where, N is the total number of documents in the corpus, and $|\{d \in D: t \in d\}|$ is the number of documents where the word *t* is present. If the term is not in the corpus, then it will lead to a divide-by-zero error, hence it is also common to adjust the denominator to1+$|\{d \in D: t \in d\}|$.

The major advantage of processing a document based on the document frequencies [1,4] is that it eliminates the overhead of processing the entire document. Since TF and IDF directly correspond to the importance of the words in the document, the resultant set of data will provide a basis for eliminating documents that have very low relevance to the actual domain being analyzed.

Integration of results

At this stage, the processing of results pertaining to independent data sets is complete and the results are available as independent entities. The result integration phase deals with combining these entities in a meaningful and consistent manner. A result set dictionary is created; containing not only the resultant data, but also the metadata pertaining to the source of the data is created.



```
Keyword Phrase: ICC World cup 2015
[{'datasource': 'Google', 'data': 'Follow the <b>Cricket World Cup</b>, 14 Feb-2
8 Mar, <b>2015</b> in Australia and New \nZealand. Official source of tickets, n
ews, schedules, video highlights &amp; photos.', 'title': 'Cricket World Cup 201
5 - ICC Cricket | Official Website'}, {'datasource': 'Google', 'data': 'ICC <b>C
ricket World Cup 2015</b> with live cricket scores and the latest news and \nfea
tures throughout the series.', 'title': 'ICC Cricket World Cup 2015 | Cricket ne
ws, live scores, fixtures ...'}, {'datasource': 'Google', 'data': 'Watch ICC <b>
Cricket World Cup 2015</b> live cricket streaming online and get live \nscore up
dates which let you know how your favourite team is performing, on\xa0...', 'tit
le': 'ICC Cricket World Cup 2015 Live Streaming &amp; Live Score on ...'}, {'dat
```

Fig 3: A Snapshot of the dictionary created for the keyword search 'ICC World cup 2015'

The dictionary contains keys pertaining to the dataset being used. The metadata considered in the current application includes title, data source, timestamp, category, contributor, keywords and source URL. Not all fields are applicable for every dataset. Hence the applicable metadata are added as entries to the final dictionary set.

Duplicate elimination and ranking

The content retrieved by various sources also has a high probability of being taken from the same source or being very similar [5, 16]. These similarities can be identified by using their similarity scores [6, 17]. Various methods are available for calculating similarities between documents, this paper presents methods that calculates the directional similarities of text and finally the total document similarity [7, 18].

A directional similarity score sim_d (T_i, T_j) is computed from a text T_i to a second text T_j(Eq.3). Therefore, for each word W_i in $T_{i,}$ its best-matching counterpart in T_j is required (maxSim($W_{i,}$ T_j)). The similarity scores of all these matches are summed up and weighted according to their inverse document frequency, and then they are normalized. The final document-level similarity is the average of applying this strategy in both directions, from T_i to T_j and vice versa (Eq. 4)

$$sim_d\left(T_i, T_j\right) = \frac{\sum_{w_i} maxSim\left(w_i, T_j\right).idf\left(w_i\right)}{\sum_{w_i} idf\left(w_i\right)} \qquad (3)$$

$$sim\left(T_i, T_j\right) = \frac{1}{2}\left(sim_d\left(T_i, T_j\right) + sim_d\left(T_j, T_i\right)\right) \qquad (4)$$

The similarity scores are compared and in document pairs *(i,j)* exhibiting 90% similarities, one of the document is eliminated to reduce the total result set. Since the major concentration of this method is to present only the precise results, it becomes mandatory to eliminate data that are redundant and would be of very less usage to the user.

The final list of candidate results are passed to the ranking section. The ranking section operates on the properties associated with the data, rather than the actual data being analyzed. It uses the weighted sum method [8,9] to determine the ranks of each of the results. Every property is assigned weights and a sum of these weights are used. If a data source contains timestamp value, then it is given higher priority. If multiple data sources contain timestamp, then the most recent data is provided the highest weight. This process of weight calculation is flexible depending on the properties present in the data. The initial weight values are assigned by the developer. The system is designed as a learning system, hence the initial weights defined are not constant and they tend to vary according to the user's perspective.

Property weight modification using user feedback

The final list of ranked candidates are presented and user's feedback is obtained, defining their preferred result. Reinforcement learning is used to improve the results. Reinforcement learning is an area of machine learning that is concerned with the activities to be carried out by software agents in order to maximize the cumulative reward. The weights pertaining to the source of the result is incremented if the result does not occupy the initial position. User's preference to a particular data source is hence reinforced, which makes accurate result predictions possible.

RESULTS AND DISCUSSION

The process of inconsistency identification and resolution is performed by retrieving results using API's and also using the available data sources, and is coded in Python. New York Times API and Google API were used to retrieve web based results and the remaining results were obtained from the created repository. Data obtained from the New York Times API is retrieved as JSON file, while that of Google is in the form of key-value pairs. Results obtained from these varied data sources are passed to the wrappers and the final dictionary is created for processing.

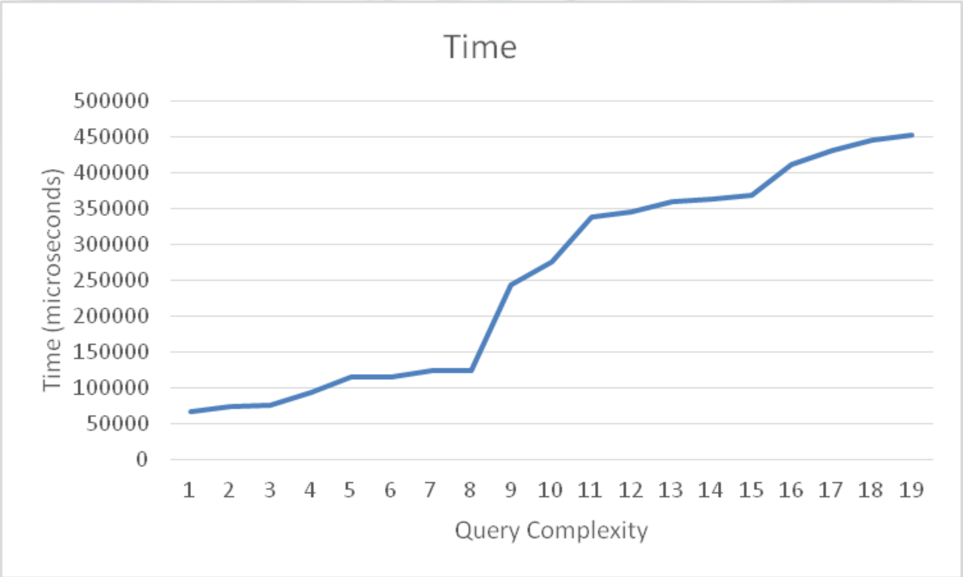


Fig 4: Time Taken Vs. Query Complexities

Figure 4 shows the time taken for result processing of varying query complexities. The queries were constructed in terms of increasing complexities. The query strings involve single word query strings to multiple word, ambiguous strings. Levels of complexities range from 1 to 19. The initial complexity levels includes clear single strings and slightly ambiguous single word queries. The mid complexity levels include multiple word queries with clear meanings to slightly ambiguous meanings. The high level complexity queries include multiple words with highly ambiguous words with the inclusion of common words.

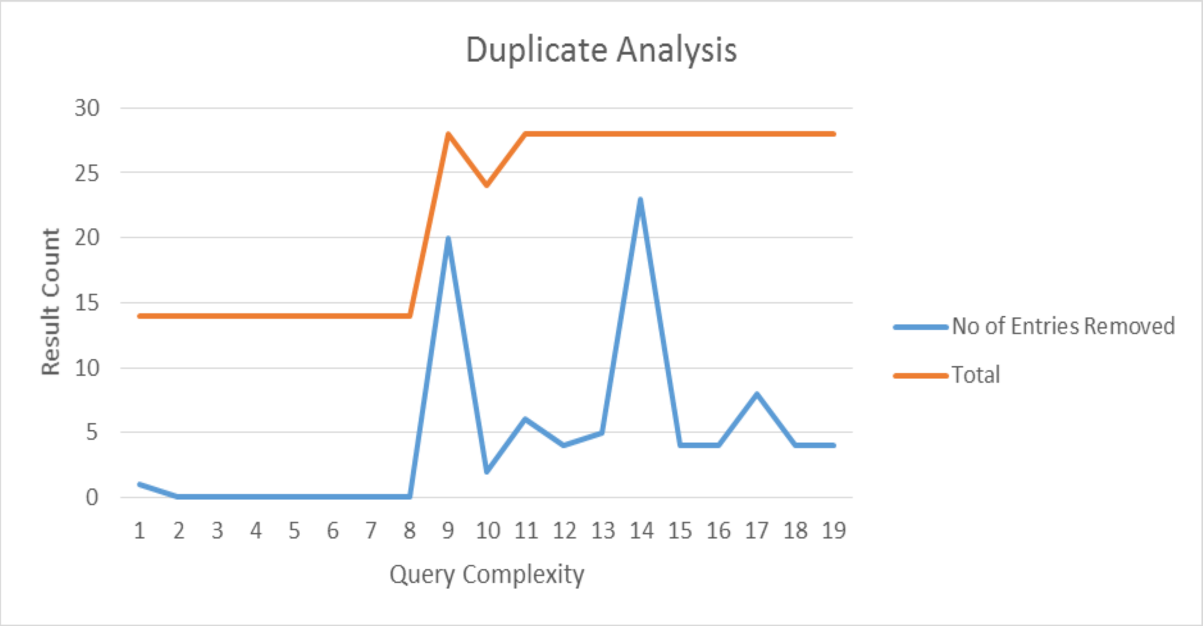


Fig 5: Duplicate Analysis

Figure 5 presents the ratio of total number of entries present in the combined dictionary and the number of entries eliminated as duplicates. The gap between the two lines indicates the size of the shortlisted candidates. It can be seen that in the initial complexity levels, the presence of duplicates is low, as the complexities increase, duplicate entries can be observed in the resultant data set.

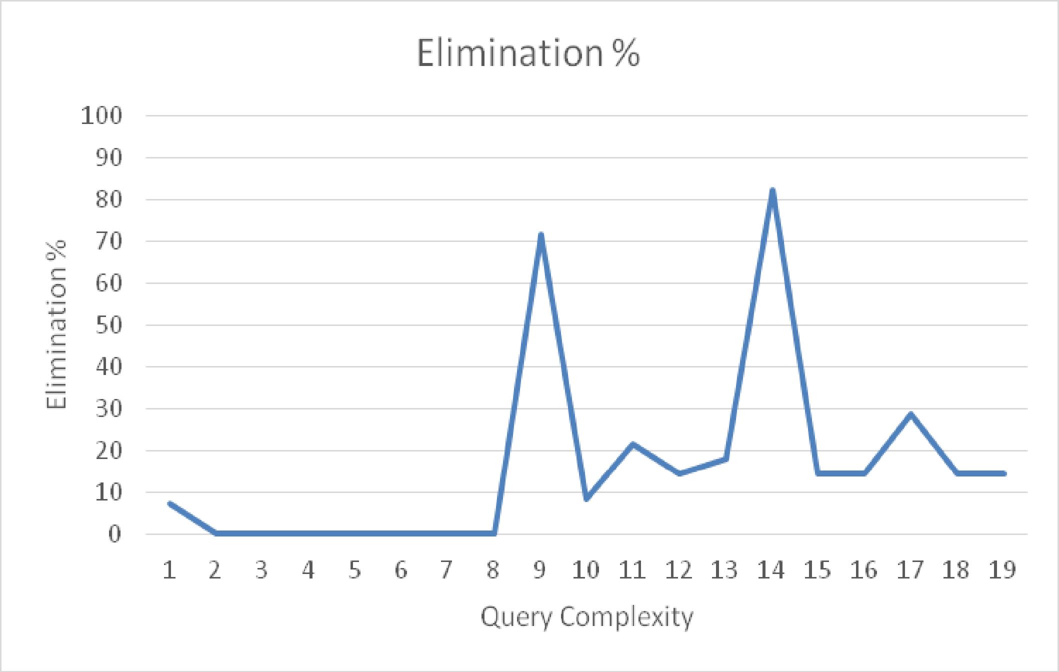


Fig 6: Elimination %

Figure 6 shows the percentage of entries eliminated from the total data. The levels of eliminations are low in events of very low and very high query complexities, while it spikes up in the mid-level complexity areas. Figure 7 shows the level of conflict data returned by the query.

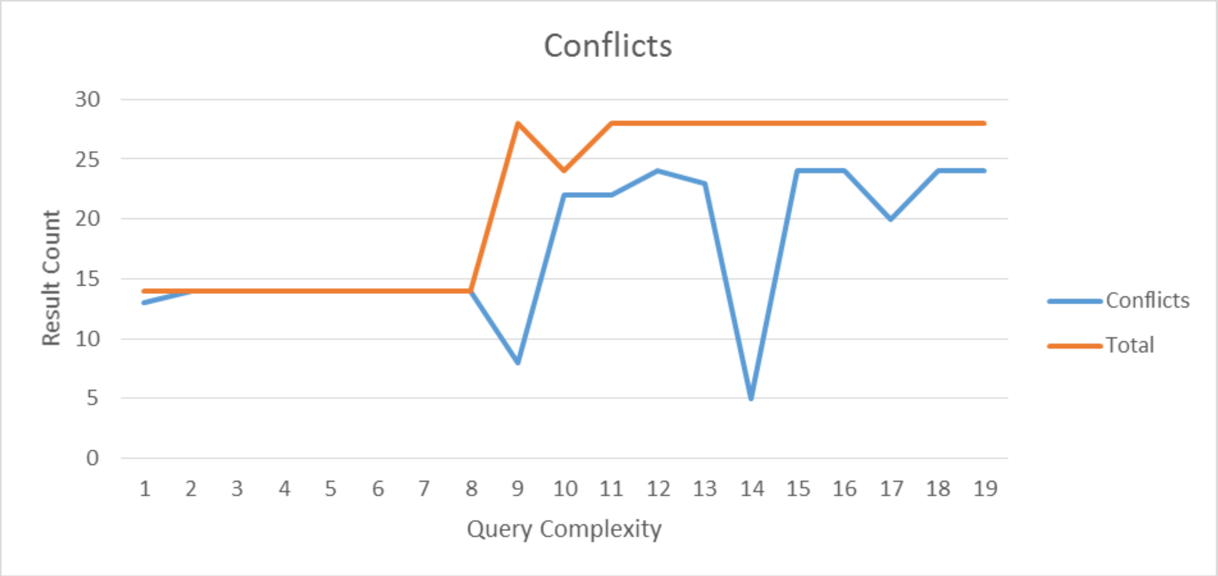


Fig.7: Conflicts

CONCLUSION

The feedback based conflict identification and resolution using duplicate elimination and ranking techniques presented in this paper provides a learning architecture that automatically detects duplicates and eliminates them. The other remaining elements are termed as conflicts. These conflicts are ranked depending on their properties and the final ranked results are provided to the user for feedback. The feedback mechanism is used to reinforce and tune the architecture for effective predictions. The advantage of this method is that it provides a generic method that can incorporate any number or type of data sets, by providing the appropriate wrapper classes. The usage of reinforcement learning method will help the architecture learn the dynamicity of the data sets being used. Future enhancements include developing a universal wrapper that works with any type of data set, when provided with appropriate ontologies.

REFERENCES

[1] Ramos, Juan. 2003. Using tf-idf to determine word relevance in document queries. Proceedings of the first instructional conference on machine learning.

[2] Salton, G., Buckley, C. 1988. Term-weighing approache sin automatic text retrieval. In Information Processing & Management, 24(5): 513-523.

[3] Berger, Adam, Caruana, R. Cohn, D. Freitag, D. and Mittal, V. 2000. Bridging the Lexical Chasm: Statistical Approaches to Answer Finding. In Proc. Int. Conf. Research and Development in Information Retrieval, 192-199.

[4] Berger, A. and Lafferty, J. 1999. Information Retrieval as Statistical Translation. In Proceedings of the 22nd ACM Conference on Research and Development in Information Retrieval (SIGIR.99), 222-229.

[5] Zesch, Torsten, D. and Gurevych, I. 2012. Text reuse detection using a composition of text similarity measures. Proceedings of COLING. Vol. 1.

[6] Mihalcea, Rada, Corley, c. and Strapparava, C. 2006. Corpus-based and knowledge-based measures of text semantic similarity. AAAI. Vol. 6.

[7] Mihalcea, R., Corley, C., and Strapparava, C. 2006. Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In Proceedings of the 21st National Conference on Artificial Intelligence, pages 775–780, Boston, MA, USA.

[8] Fishburn, P.C. 1967. Additive Utilities with Incomplete Product Set: Applications to Priorities and Assignments. Operations Research Society of America (ORSA), Baltimore, MD, U.S.A.

[9] Triantaphyllou, E. 2000. Multi-Criteria Decision Making: A Comparative Study. Dordrecht, the Netherlands: Kluwer Academic Publishers (now Springer). p. 320.ISBN 0-7923-6607-7.

[10] Dong, X., Naumann F. 2009. Data Fusion – Resolving Data Conflicts for Integration . VLDB 09. 24-28.

[11] Siraj, Sajid, Mikhailov, L. and Keane, j. 2015. Contribution of individual judgments toward inconsistency in pairwise comparisons. European Journal of Operational Research 242.2,557-567.

[12] Thimm and Matthias. 2014. Towards Large-Scale Inconsistency Measurement. KI 2014: Advances in Artificial Intelligence. Springer International Publishing. 195-206.

[13] Chien, Chian, B. and Hsueh, Y. 2014. Layered Context Inconsistency Resolution for Context-Aware Systems. Modern Advances in Applied Intelligence. Springer International Publishing. 446-455.

[14] Deagustini, David, C.Martínez, M. Marcelo, Falappa, A. and Simari, G. 2014. Improving Inconsistency Resolution by Considering Global Conflicts." In Scalable Uncertainty Management, pp. 120-133. Springer International Publishing.

[15] Fahad, Muhammad, Nejib Moalla, and Bouras, A. 2012. Detection and resolution of semantic inconsistency and redundancy in an automatic ontology merging system. Journal of Intelligent Information Systems 39.2: 535-557.

[16] Zesch, Torsten, D. and Gurevych, I. 2012. Text reuse detection using a composition of text similarity measures. Proceedings of COLING. Vol.

[17] Mihalcea, Rada, Corley, C. and Strapparava, C. 2006. Corpus-based and knowledge-based measures of text semantic similarity. AAAI. Vol. 6.

[18] Hatzivassiloglou, Vasileios, Judith L. Klavans, and Eskin, E. 1999. Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning. Proceedings of the 1999 joint sigdat conference on empirical methods in natural language processing and very large corpora.

Author' biography with Photo

I. Carol is pursuing doctor of philosophy in Department of Computer Science, St. Joseph's College, (Autonomous), Tiruchirappalli, Tamil Nadu, India. He received his M. Phil degree from St. Joseph's College, Tiruchirappalli. He received his MCA degree from St. Joseph's College, Tiruchirappalli. He has published many research articles in the International conferences and journals. His area of interest is Data mining and Web mining.

Dr. S. Britto Ramesh Kumar is working as Assistant Professor in the Department of Computer Science, St. Joseph's College (Autonomous), Tiruchirappalli, Tamil Nadu, India. He has published many research articles in the National/International conferences and journals. His research interests include Data Mining, Web Mining, and Mobile Networks.