Learning to Handle Inconsistency for Multi-Source Integration

Sheila Tejada Craig A. Knoblock Steven Minton

University of Southern California/ISI

4676 Admiralty Way, Marina del Rey, California 90292

{tejada,knoblock,minton}@isi.edu,

(310) 822-1511 x799

Abstract

Many problems arise when trying to integrate information from multiple sources on the web. One of these problems is that data instances can exist in inconsistent formats across several sources. An example application of information integration is trying to integrate all the reviews of Los Angeles restaurants from Yahoo's Restaurants webpage with the current health rating for each restaurant from the LA County Department of Health's website. Integrating these sources requires determining if they share any of the same restaurants by comparing the data instances from both sources (**Figure 1**). Because the instances can be in different formats, e.g. the restaurant "Jerry's Famous Deli" from Yahoo's webpage can appear as "Jerry's Famous Delicatessen" in the Dept. of Health's source, they can not be compared using equality; but must be judged according to similarity.

Name	Address	Phone
1. Jerry's Famous Deli	342 Beverly Blvd,	(310)302-5319
Jerry's Famous Delicatessen	342 Beverly Boulevard,	310-302-5319
2. CPK	65 La Cienga Blvd,	310-987-8923
California Pizza Kitchen	65 La Cienga Blvd,	310-987-8923
3. The Beverly	302 MLK Blvd,	213-643-2154
Cafe Beverly	302 Martin Luther King Jr. Boulevard,	645-4278

Figure 1*: Matched Restaurant*

Once the matching objects have been determined, this information about which objects are mapped together can be stored in the form of a mapping table, or as a mapping function, so that these sources can be properly integrated in the future. The goal of this research [1] is to be able to create these mapping constructs so that an information broker, like Ariadne [2], can use it to properly integrate data from inconsistent sources in an intelligent and efficient manner.

We have currently developed a semi-automated tool that is able to create these mapping constructs. This tool is composed of three main components (**Figure 2**). The first component, the match generator, compares all of the shared attributes of the sources in order to determine which objects are matched, (**Name** with **Name**, **Address** with **Address, Phone** with **Phone**). To determine the similarity between the instances we have developed general domain independent transformation rules to recognize transformations like substring, acronym, and abbreviation. For example, "Deli" is a substring transformation of "Delicatessen." This component computes the set of possible matched pairs using these general transformation rules and calculates a similarity score for each of the attributes of each matched pair.

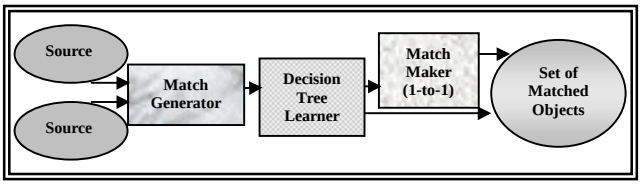

Figure 2: *Mapping Tool Components*

The second component is a decision tree learner which takes, as input, a subset of the possible matched pairs that have been labeled as matched or not matched by the user. From this training set the learner can discover which attribute or combination of attributes are most important for matching objects by determining the thresholds for the probabilistic score of these attributes. For this restaurant example, the learner could discover that in order for two objects to be matched, the similarity score for the attribute **Name** must be greater than 0.8. The learned decision tree is then used to classify the remaining set of possible matched pairs. The third component is a one-to-one match maker which enforces the one-to-one relationship that might exist in the application domain. For our future work we are adding an active learning or query by committee technique in order to reduce the size of the training set [3].

References

[1] Tejada, et. al. Handling Inconsistencies for Multi-Source Integration. Proceedings of the 15th National Conference on Artificial Intelligence, Madison, WI, 1998.

[2] Knoblock, et. al. Modeling Web Sources for Information Integration. Proceedings of the 15th National Conference on Artificial Intelligence, Madison, WI, 1998.

[3] Naoki Abe and Hiroshi Mamitsuka. Query Learning Strategies Using Boosting and Bagging. Proc. of the 15th International Conference on Machine Learning, July 1998.