Instance Based Schema Matching Framework Utilizing Google Similarity and Regular Expression

Osama A. Mehdi, Hamidah Ibrahim and Lilly Suriani Affendey
Faculty of Computer Science and Information Technology, 43400, Serdang, Malaysia

Keywords: Schema Matching, Instance based Schema Matching, Google Similarity, Regular Expression.

Abstract: Schema matching is the task of identifying correspondences between schema attributes that exist in different schemas. A variety of approaches have been proposed to achieve the main goal of high-quality match results with respect to precision (*P*) and recall (*R*). However, these approaches are unable to achieve high quality match results, as most of these approaches treated the instances as string regardless the data types of the instances. As a consequence, this causes unidentified matches especially for attribute with numeric instances which further reduces the quality of match results. Therefore, effort still needs to be done to further improve the quality of the match results. In this paper, we propose a framework for addressing the problem of finding matches between schemas of semantically and syntactically related data. Since we only fully exploit the instances of the schemas for this task, we rely on strategies that combine the strength of Google as a web semantic and regular expression as pattern recognition. To demonstrate the accuracy of our framework, we conducted an experimental evaluation using real world data sets. The results show that our framework is able to find 1-1 schema matches with high accuracy in the range of 93% - 99% in terms of precision (*P*), recall (*R*), and F-measure (*F*).

1 INTRODUCTION

Schema matching is the problem of finding correspondences between attributes of two schemas that are heterogeneous in format and structure (e.g., relation attributes or XML tags). This basic problem needs to be solved in many database application domains, such as data integration, E-business and data warehousing (Bernstein et al., 2011). Performing schema matching from different sources is not trivial as these sources are developed independently by different developers, thus leading to differences in terms of structure, syntactic and semantics of the schema attributes. Schema matching attempts to measure the similarities between schema attributes by considering the schema information which includes the element name (schema name, attribute name), description, data type and schema structure. There are several approaches that have been developed for handling schema matching (Doan et al., 2000). Interested readers may refer to the following surveys (Rahm and Bernstein, 2001; Shvaiko and Euzenat, 2005; Blake, 2007; Bernstein et al., 2011) and books (Euzenat and Shvaiko, 2007; Hai, 2007; Bellahsene et al., 2011).

In (Rahm and Bernstein, 2001), the existing schema matching approaches are classified into the following categories: (i) Schema Level Approaches - that use schema information, (ii) Instance Level Approaches - that use the data instances as source for finding the correspondences of schema attributes, and (iii) Hybrid Approaches - that combine information extracted from both the schema and the instances. However, in some cases it may not be possible to use the schema information. For instance, depending on attribute name does not always work properly since database designers tend to use *compound nouns*, *abbreviations* and *acronyms*. Although lexical annotation helps in associating meaning to attribute names, however, the performance of lexical annotation methods on real world schemata suffers from the abundance of *non-dictionary words* such as *compound nouns*, *abbreviations* and *acronyms*. The result of lexical annotation is strongly affected by the presence of these *non-dictionary words* (Cortez et al., 2010). Hence, exploring the instances can give an accurate

A. Mehdi O., Ibrahim H. and Affendey L..
Instance Based Schema Matching Framework Utilizing Google Similarity and Regular Expression.
DOI: 10.5220/0004990102130222
In *Proceedings of 3rd International Conference on Data Management Technologies and Applications* (DATA-2014), pages 213-222
ISBN: 978-989-758-035-2

characterization of the actual contents of schema attributes (Rahm and Bernstein, 2001; De Carvalho et al., 2013).

By analysing the instance based schema matching approaches, we observed that neural network, machine learning, theoretic information discrepancy and rule based have been utilized (Kang and Naughton, 2003; Chua et al., 2003; Bilke and Naumann, 2005; Liang, 2008; Kang and Naughton, 2008; Dai et al., 2008). The goal of these approaches is to discover correspondences between schema attributes whereby instances including instances with numeric values are treated as strings. This prevents discovering common patterns or performing statistical computation between the numeric instances (De Carvalho et al., 2013). As a consequence, this causes unidentified matches for numeric instances and further reduces the quality of match results. Furthermore, textual similarity also is not the best alternative for numeric instances (e.g., page number, year, phone number, price, quantity, etc.) (De Carvalho et al., 2013; Cortez et al., 2010). Thus, for instance level approaches, specific strategies for identifying existing instance patterns must be deployed.

In this paper, we propose a framework for instance based schema matching that aims at finding the correspondences between schema attributes of two semantically and syntactically related data. Since we only explore the instances, we rely on matching strategies that are based on Google similarity (Cilibrasi and Vitanyi, 2007) and regular expression (Friedl, 2006; Liu et al., 2012) to find the correspondences of schema attributes. As pointed out by (Doan and Halevy, 2005; Li and Clifton, 2000), there are different types of matching algorithms being applied in this area. However, this problem is still a research hotspot in order to further improve the accuracy of schema matching. Thus, our framework is a step forward towards solving this problem. The reason for utilizing Google similarity and regular expression is that Google similarity uses the World Wide Web as database and Google as search engine. Whereas regular expressions are an efficient way to describe text through pattern (format) matching and provide an efficient way to identify text. In addition, regular expression is relatively inexpensive and does not require training as in learning-based techniques. It can provide a quick and concise method to capture valuable knowledge (Doan and Halevy, 2005).

In summary, the main contribution of this paper is a framework which: (1) uses only the instances to find matches between schema attributes (1-1 matches) and (2) relies on matching strategies that are based on Google similarity and regular expression to find the matches.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 presents the proposed framework of instance based schema matching. In Section 4, the evaluation metrics and the results are presented and discussed. Finally, Section 5 draws the conclusions and points out some future work directions.

2 RELATED WORK

Instance based schema matching examines instances to determine corresponding schema attributes. It represents a substitutional choice for schema matching (Rahm and Bernstein, 2001; Bernstein et al., 2011). Even when substantial schema information is available, considering instances can complement schema based approaches with additional insights on the semantics and contents of schema attributes and can be beneficial in uncovering wrong interpretation of schema information, i.e. it would be helpful to disambiguate between schema level matches by matching the attributes whose instances are syntactically and semantically more similar. Neural network, machine learning, information theoretic discrepancy and rule based are approaches used for instance based schema matching.

Neural network is able to obtain the similarities among data directly from their instances and empirically infer solutions from data in the absence of prior knowledge for regularities. Neural network is employed to cluster similar attributes, whose instances are uniformly characterized using a feature vector of constraint based criteria. For instance based schema matching, the Back Propagation Neural Network (BPNN), which can acquire and store a mass of mappings between input and output, is ideal. However, neural network can be viewed as specific tool since it is trained based on domain-specific training data. It can only be used to resolve problems associated with that domain (Li et al., 2000). Furthermore, neural network approaches (Li and Clifton, 1994; Li and Clifton, 2000; Yang et al., 2008; Li et al., 2005) for instance based schema

matching achieved precision (*P*), recall (*R*), and F-measure (*F*) in the range of 65% - 96%.

Solutions that are based on machine learning generally employ methods such as Naïve Bayesian classification to enhance the accuracy of schema based matching. Learning-based solutions require a training data set of correct matches that may require a large training data set to determine the correct matches. Several approaches have been proposed (Doan et al., 2001; Berlin and Motro, 2001; Feng et al., 2009) that employ machine learning techniques to first learn the instance characteristics of the matching or non-matching attributes and then use them to determine if a new attribute has instances with similar characteristics or not. However, the precision (*P*), recall (*R*), and F-measure (*F*) achieved by these approaches are in the range of 66% - 92%.

Many approaches have applied the notion of information theoretic discrepancy such as mutual information and distribution values (Liang, 2008; Kang and Naughton, 2003; Kang and Naughton, 2008; Dai et al., 2009). The main advantages of applying an information theoretic discrepancy approach are that its skilfulness and lack of constraints. However, approaches of information theoretic discrepancy need some probabilities of overlapping in the values being compared. Furthermore, information theoretic discrepancy approaches for instance based schema matching achieved precision (*P*), recall (*R*), and F-measure (*F*) in the range of 45% - 92%.

Rule based approaches enjoy many benefits. The first benefit of using rule based would be, low cost and also no requirement for training as in learning-based techniques. The second benefit, its quick and concise method to capture valuable user knowledge about the domain. Finally, rule based approaches for instance based schema matching (Chua et al., 2003; Bilke and Naumann, 2005) achieved precision (*P*), recall (*R*), and F-measure (*F*) in the range of 72% - 87%.

3 THE PROPOSED FRAMEWORK OF INSTANCE BASED SCHEMA MATCHING

The major tasks of the proposed framework are to analyse the instances of the schemas and determine correspondences of attributes between schemas. Figure 1 illustrates the proposed framework of instance based schema matching which consists of two main components, namely: (i) Pre-Processing and (ii) Instance matching.

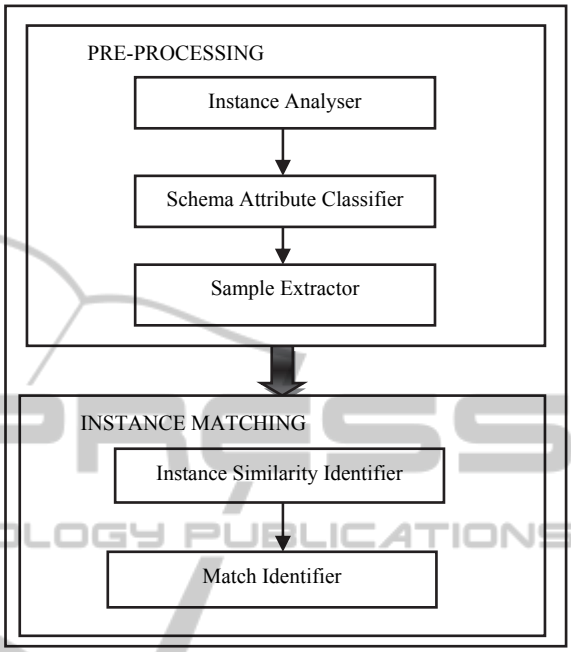

Figure 1: The proposed framework of instance based schema matching.

3.1 The Sub-components of the Pre-processing

The Pre-Processing comprises of three sub-components, namely: Instance Analyser, Schema Attribute Classifier, and Sample Extractor. These sub-components are further explained below:

3.1.1 Instance Analyser

This sub-component analyses the instances of each attribute in a schema in an attempt to identify the data type of the attribute. The characters of a value that represent a randomly selected instance of each attribute are analysed to determine which data type the attribute belongs to. There are three types of data type, namely: *alphabetic*, *numeric* and *mix* (alphabetic, numeric and special characters). The *alphabetic* data type is for attributes whose instances consist of only alphabetic characters ([A...Z, a…z]). The *numeric* data type is for attributes whose instances consist of only digit characters ([0…9]), whereas the *mix* data type is for attributes whose instances consist of combination of alphabetic, digit

and special characters (e.g [-, /, \, .,]). Whereas, it's possible to use data type constraints in our framework by mapping the data types that we are producing to the equivalent data type constraints. However, using data type constraints might leads our framework to be unable to detect the correct correspondences between the attributes that have the same data type of instances. For instance, database designers tend to use string data type for numeric attributes that do not need any kind of calculating. Hence, if we used data type constraints we may classify such attributes in unrelated class for matching.

3.1.2 Schema Attribute Classifier

This sub-component classifies the attributes of the schemas into classes based on the data type of each attribute that has been derived from the Instance Analyser sub-component. In other words, attributes of the same data type are gathered in the same class. The main aim of this phase is to reduce the number of possible comparisons that needs to be performed during the matching process. The maximum number of classes created for each schema is based on the number of data types that has been determined from the Instance Analyser sub-component.

3.1.3 Sample Extractor

This sub-component intends to extract instances from the initial table of data set based on the optimal sample size and populate them into a table. Each table consists of a number of attributes from different schemas which have the same data type. The optimal sample size represents the size of samples that achieves the acceptable results of instance based schema matching in terms of precision (*P*), recall (*R*) and F-measure (*F*). The optimal sample size has been chosen through a set of experiments. More details about these experiments are discussed in Section 4. The purpose of this sub-component is to reduce the number of comparisons between the instances that further reduce the processing time of instance based schema matching.

3.2 The Sub-components of the Instance Matching

The instance matching component encompasses two sub-components, namely: Instance Similarity Identifier and Match Identifier.

3.2.1 Instance Similarity Identifier

This is the major component in instance matching which aims at determining whether the instances of attributes that are in the same class have similarities. To achieve this, two strategies are adopted to find the similarities between the attributes that are in the same class. The strategies are: Google similarity for calculating the semantic similarity and regular expression for the syntactic matching.

Google Similarity

For measuring the semantic similarity we have adopted Google similarity that has been introduced by (Cilibrasi and Vitanyi, 2007; Cilibrasi and Vitanyi, 2004). Google similarity uses the World Wide Web as a database and Google as a search engine. Google's similarity of words and phrases from the World Wide Web uses Google page counts, as shown in equation (1).

$$\mathrm{GSD}(x, y) = \frac{max(\log f(x), \log f(y)) - \log f(x, y)}{\log M - min(\log f(x), \log f(y))} \quad (1)$$

where $f(x)$ is the number of Google hits for the search term x, $f(y)$ is the number of Google hits for the search term y, $f(x, y)$ is the number of Google hits for both terms x and y together, and M is the number of web pages indexed by Google. The World Wide Web is the largest database on earth and the context information entered by millions of independent users averages out to provide automatic semantics of useful quality. The Google similarity calculates the semantic similarity score for the attributes with alphabetic data type that comprises instances consisting of only alphabetic characters ([A...Z, a…z]). For instance, if we want to search for a given term in the Google web pages, e.g. "Msc", we will get a number of hits that is 127,000,000. This number refers to the number of pages where this term is found. For another term, "Phd", the number of hits for this term is 50,600,000. Furthermore, if we search for those pages where both terms " Msc" and "Phd" are found, that gives us 36,100,000 hits. Consequently, we can use these numbers of hits for the terms "Msc", "Phd" and both terms together in addition to the number of pages indexed by Google, which is around 3,000,000,000 in the equation (1). The equation produces the similarity degree between the two terms "Msc" and "Phd" as follows:

$$GSD(Msc, Phd) = 0.31$$

Regular Expression

Regular expression (known as regexes) is a way to describe text through pattern (format) matching and provide an easy way to identify text. Regular expression is a language used for parsing and manipulating text (Kleene, 1051). Furthermore, regular expression is a string containing a combination of normal characters and special metacharacters or metasequences (*, +, ?). Table 1 shows the most common metacharacters and metasequences in regular expression that are used in our work (Friedl, 2006).

Table 1: The most common metacharacters in regular expression.

Metacharacter	Name	Matches
d\	Digit	Matches a digit
s\	Whitespace	Matches whitespace
[a-z, A-Z]	A range of letters	Matches any letter in the specified range.
.	Dot	Matches any one character
[…]	Character class	Matches any one character listed
[^…]	Negated character class	Matches any one character not listed
?	Question	One allowed, but it is optional
*	Star	Any number allowed, but all are optional
+	Plus	At least one required; additional are optional
|	Alternation	Matches either expression it separates
^	Caret	Matches the position at the start of the line
$	Dollar	Matches the position at the end of the line
{X,Y}	Specified range	X required, max allowed

The attributes with *numeric* data type are attributes whose instances consist of only digit characters ([0…9]). In creating a regular expression for an attribute, the minimum and maximum values of the attribute are required. Thus, three variables have been identified, namely: *nomin*, *nomax* and *uppervalue*. Initially, *nomin* and *nomax* are assigned the minimum and maximum values of the attribute, respectively. However, in the following iterations, the value of *nomin* is changed to the last *uppervalue* + 1. The *uppervalue* is a value which is greater than the value of *nomin* and less than the value of *nomax*; and is derived based on the following conditions:

- when the *nomin's* length of digits is less than the *nomax's* length of digits, the *uppervalue* is the maximum value based on the *nomin's* length of digits and not greater than the value of *nomax*.
- when the *nomin's* length of digits is equal to the *nomax's* length of digits and the *nomin* has at least one zero digit on the right, the *uppervalue* is derived using the formula shown in equation (2). The equation (2) derives the closest *uppervalue* to the *nomax*.

$$uppervalue = (nomax - (nomax \ MOD \ Sumz * 10) - 1) \quad (2)$$

where *Sumz* returns number of zero digits on the right of the *nomin*. If the equation (2) returns an *uppervalue* which does not satisfy the condition that we have stated earlier, then the step as mentioned in (i) above is applied. Table 2 illustrates an example of the proposed idea for *numeric* data type.

Table 2: An example of numerical data type.

Iteration	Nomin	Upper Value	RegEx	Accumulated RegEx
1	7	9	[7-9]	[7-9]
2	10	99	[1-9][0-9]	[7-9]|[1-9][0-9]
3	100	119	1[0-1][0-9]	[7-9]|[1-9][0-9]|1[0-1][0-9]
4	120	123	12[0-3]	[7-9]|[1-9][0-9]|1[0-1][0-9]|12[0-3]

On the other hand, for the attributes with *mix* data type whose instances consist of combination of alphabetic, digit and special characters (e.g [-, /, \, .,]), we divide the instances into a set of sub-tokens. Each sub-token is a sequential set of characters of a particular data type. Then, a regular expression is built for each sub-token of the instance. Finally, the regular expressions of each sub-token are combined as the regular expression of the instance. Table 3 illustrates an example of the proposed idea for *mix* data type.

Table 3: An example of the mix data type.

Instance	Regular Expression
255 Courtland	d\+\s[a-z, A-Z]+
589/265/954	d\+/d\+/d\+

3.2.2 Match Identifier

This sub-component attempts to determine whether the identified match between attributes from different schemas are the same real world entity.

For numeric and mix data types the same process is performed, as they use the concept of regular expression. First of all, this sub-component specifies a match by matching the regular expression of the instances of the source schema attribute derived from the previous sub-component against sample of instances of each target schema attribute. Then, it counts the number of instances of the target schema attribute that matchs the regular expression. Next, this sub-component identifies the percentage similarity for each attribute of the target schema with the regular expression. The percentage similarity is identified by dividing the number of instances that matchs the regular expression with number of instances of target schema attribute. The highest percentage is then identified and if it is greater than 50% , then we can say that these attributes are correspond to each other.

On the other hand, for the alphabetic data type this sub-component uses the list of similarity scores derived from the previous sub-component (utilizing Google similarity) with a predefined threshold value. In our work the threshold value is set to 60 as used by previous work (Khan et al., 2011). The list of similarity scores contains the average similarity score for each attribute of the source schema with each attribute of the target schema. Hence, this sub-component identifies the highest score of similarity achieved between the attribute of the target schema and the attribute of the source schema. If the highest score is equals to or greater than 50%, then these attributes are said to correspond to each other.

4 EVALUATION

4.1 Data Set

We used real-world data sets from two different domains: Restaurant and Census, both of which are available online (Restaurant, 2014; Census, 2014). Table 4 shows the characteristics of the data sets. In our experiments we created two sub-tables by randomly selecting the attributes from the original table of the data sets. The number of attributes of each sub-table is equal to the number of attributes of the original table. However, these attributes might occur in different sequence and the same attributes might be selected more than once. These sub-tables were populated with instances taken randomly from the original table of the data sets. The number of instances of both sub-tables is different to represent real world cases. We pretended that these sub-tables were two different tables that needed to have their schemas matched (Liang, 2008; Kang and Naughton, 2003; Kang and Naughton, 2008).

Table 4: The Characteristics of the data sets.

Data Set	Restaurant	Census
Number of Attributes	5	11
Alphabetic Attributes	Name, Type of Food and City	Workclass, Education, Relationship, Race, Sex, Marital status, and Native-country
Numeric Attributes	-	Age, Fnlwgt, Education-num and Capital-gain
Mix Attributes	Address and PhoneNumber	-
Number of Records	864	4320
Number of Instances	32561	358171

4.2 Measurements

The evaluation metrics considered in this work are precision (*P*), recall (*R*) and F-measure (*F*) shown in equations (3), (4) and (5), respectively. It is based on the notion of true positive, false positive, true negative, and false negative.

- *True positive* (*TP*): The number of matches detected when it is really matches.
- *False positive (FP*): The number of matches detected when it is really non-match.
- *True negative* (*TN*): The number of non-matches detected when it is really non-match.
- *False negative* (*FN*): The number of non-matches detected when it is really matches.

$$Precision = |TP| / |TP| + |FP| \quad (3)$$

$$Recall = |TP| / |TP| + |TN| \quad (4)$$

$$F\text{-}measure = 2 * Precision * Recall / Precision + Recall \quad (5)$$

For each table, we kept the number of attributes to 11 and 5 for Census and Restaurant data sets, respectively. We repeated each experiment 5 times, measured the precision (*P*), recall (*R*) and F-measure (*F*) and averaged these results.

4.3 Result

We have conducted two analyses. They are (i) Analysis 1 which aims at identifying the optimal sample size of tuples and (ii) Analysis 2 which aims

at comparing the performance of our proposed framework to that of the previous work with respect to precision (*P*), recall (*R*) and F-measure (*F*). The details of each analysis are presented in the following subsections.

4.3.1 Analysis 1

In this analysis, we present the experiments of selecting the optimal sample size of tuples, which represents the size of samples that achieves acceptable results in terms of precision (*P*), recall (*R*), and F-measure (*F*). The optimal sample size is the number of tuples that are used during the sub-component of Sample Extractor of instance based schema matching. For this analysis several experiments have been conducted. The experiments have been designed in such a way that each experiment will use different size of samples starting from 5% of the actual table size. The size of samples is increased either 5% or 10% in the subsequent experiments. The experiments are ended when the precision (*P*), recall (*R*) and F-measure (*F*) are at least 96% which is at least equal to the best results reported in the previous work (Yang et al, 2008). From this analysis we found that when the size of samples reached 50% the results of precision (*P*), recall (*R*) and F-measure (*F*) are better than the previous work. Table 5 illustrates the size of samples considered in each experiment.

Table 5: Size of samples for each experiment.

Experiment	Size of Samples
Experiment 1-1	5%
Experiment 1-2	10%
Experiment 1-3	15%
Experiment 1-4	20%
Experiment 1-5	25%
Experiment 1-6	30%
Experiment 1-7	40%
Experiment 1-8	50%

The experiments are labeled as Experiment 1-1, Experiment 1-2, Experiment 1-3, Experiment 1-4, Experiment 1-5, Experiment 1-6, Experiment 1-7 and Experiment 1-8. These eight experiments used the same data sets. To ensure that the results are consistent through of the experiments. Each experiment is run 5 times.

Result of Analysis 1

We reported the precision (*P*), recall (*R*) and F-measure (*F*) for the experiments 1-1, 1-2, 1-3, 1-4, 1-5, 1-6, 1-7 and 1-8 as shown in Table 6 and Table 7. While Figure 2 and Figure 3 present the precision (*P*), Recall (*R*) and F-measure (*F*) for the experiments of this analysis in histogram. The percentage increases as the sample size increases. For example, the percentages are 69% and 70% for precision (*P*) and recall (*R*), respectively when the size of samples is 5%, while these percentages increased to 87% and 100% when the size of samples is 25%. Although we have mentioned that acceptable results mean the results of precision (*P*), recall (*R*) and F-measure (*F*) are at least equal to the best results as reported in previous work, however in this analysis the precision (*P*) is lower but the recall (*R*) and F-measure (*F*) are higher than those reported in the previous work (Yang et al, 2008). Compared to the results shown in Table 6 for the Restaurant data set there is a slight different in the results of Census data set as shown in Table 7. For example, when the size of samples is 5% the precision (*P*) and recall (*R*) achieved for the Restaurant data set are 69% and 70% respectively, while for the Census data set, the precision (*P*) and recall (*R*) are 61% and 80%, respectively. The precision (*P*) and recall (*R*) increased to 81% and 96% respectively when the size of samples is 25%.

The reason is due to the characteristics of Restaurant data set that consists of three attributes with *alphabetic* data type and two attributes with *mix* data types. From the results we can conclude that 50% of the actual table size is the optimal sample size that represents the number of tuples that will be used during the sub-component of Sample Extractor of instance based schema matching. Thus, we have stopped the experiments at this stage as the results achieved with the sample size of 50% outperformed the results reported in the previous works in terms of precision (*P*), recall (*R*), and F-measure (*F*).

Table 6: Results related to the restaurant data set for the eight experiments.

Experiment (EX)	Size of Samples	Precision (P)	Recall (R)	F-measure (F)
Ex 1-1	5%	69%	70%	70%
Ex 1-2	10%	78%	80%	79%
Ex 1-3	15%	80%	94%	87%
Ex 1-4	20%	83%	98%	90%
Ex 1-5	25%	87%	100%	93%
Ex 1-6	30%	86%	100%	92%
Ex 1-7	40%	86%	100%	92%
Ex 1-8	50%	89%	100%	95%

Table 7: Results related to the census data set for the eight experiments.

Experiment (EX)	Size of Samples	Precision (P)	Recall (R)	F-measure (F)
Ex 1-1	5%	61%	80%	69%
Ex 1-2	10%	70%	88%	78%
Ex 1-3	15%	74%	93%	85%
Ex 1-4	20%	76%	90%	82%
Ex 1-5	25%	81%	96%	88%
Ex 1-6	30%	86%	100%	92%
Ex 1-7	40%	91%	96%	93%
Ex 1-8	50%	97%	97%	97%

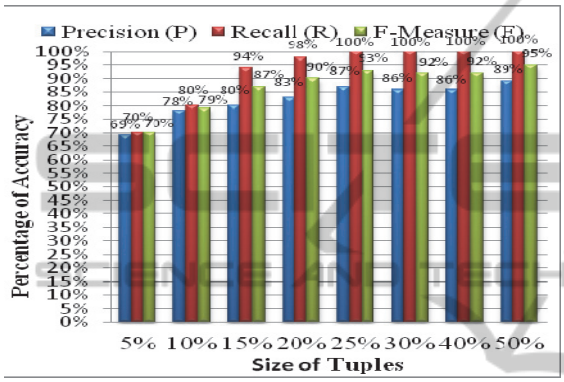

Figure 2: Percentage of *P, R* and *F* for the Restaurant data set samples.

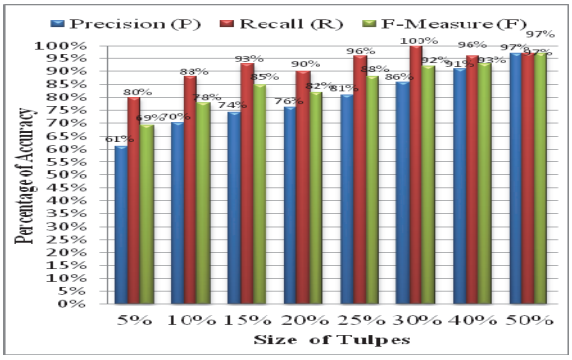

Figure 3: Percentage of *P, R* and *F* for the Census data set samples.

4.3.2 Analysis 2

In this analysis, we present the performance of our proposed framework and compare it to the previous work with respect to precision (*P*), recall (*R*), and F-measure (*F*). Figure 4 presents the results of accuracy in terms of precision (*P*), recall (*R*) and F-measure (*F*) for the proposed framework of instance based schema matching. From the results presented in Figure 4, the following can be concluded: (i) we achieved 96% for precision (*P*), 93% for recall (*R*) and 95% for F-measure (*F*) for the Restaurant data set, while with Census data set we achieved 99% for precision (*P*), 96% for recall (*R*), and 97% for F-measure (*F*). The size of samples used is 50% of the actual table size which has been identified through experiments; and (ii) our proposed framework produced high accuracy in spite of the framework considered a sample of instances instead of considering the whole instances during the process of instance based schema matching.

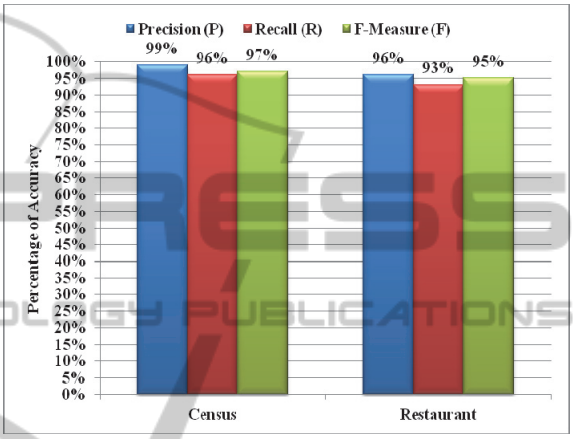

Figure 4: Matching results of Census and Restaurant data sets.

For comparison purpose, we compared our framework to the previous approach proposed by (Dai et al., 2008). We evaluated (Dai et al., 2008) approach based on the two data sets, namely: Restaurant and Census. Figure 5 and Figure 6 show the results of our proposed framework compared to the (Dai et al., 2008) in terms of precision (*P*), recall (*R*) and F-measure (*F*). From these results the approach proposed by (Dai et al., 2008) achieved low accuracy which are 66%, 68% and 67% for precision (*P*), recall (*R*), and F-measure (*F*), respectively for the Restaurant data set. While for the Census data set the approach by (Dai et al., 2008) achieved 83%, 74% and 78% for precision (*P*), recall (*R*) and F-measure (*F*), respectively. This is due to the fact that (Dai et al., 2008) approach depends on the existence of common/identical instances between the compared attributes.

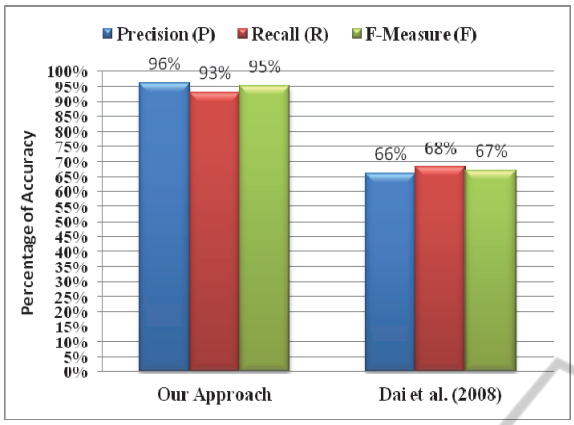

Figure 5: Matching results of the Restaurant data set.

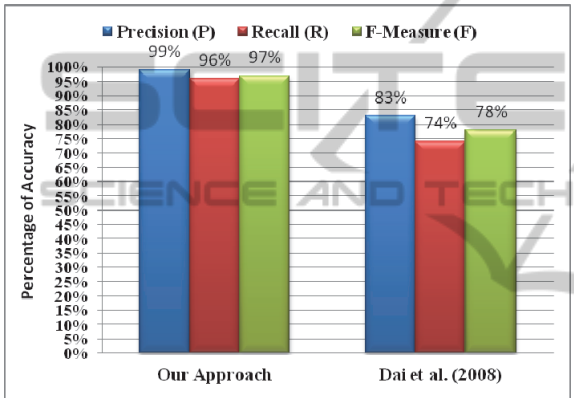

Figure 6: Matching results of the Census data set.

From these experiments, we can conclude that our proposed framework achieved better results although only a sample of instances is used instead of considering the whole instances during the process of instance based schema matching as used in the previous works (Dai et al., 2008).

5 CONCLUSIONS

In this paper, we proposed an instance based schema matching framework to identify 1-1 schema matching. Our framework rely on strategies that combine the strengths of Google as a web semantic and regular expression as pattern recognition. Our experimental results show that our framework is able to identify 1-1 matches with high accuracy in terms of precision (*P*), recall (*R*) and F-measure (*F*) although only a sample of instances is used instead of considering the whole instances during the process of instance based schema matching. In the near future, we plan to extend our framework to handle complex schema matching (n-m), since identifying complex matches is a more challenging problem.

REFERENCES

Bellahsene, Z., Bonifati, A. and Rahm, E., 2011. *Schema matching and mapping.* Springer-Verlag, Heidelberg.

Berlin, J. and Motro, A. 2001. Autoplex: Automated discovery of content for virtual databases. *Cooperative Information Systems* Springer, pp. 108-122.

Bernstein, P.A., Madhavan, J., and Rahm, E., 2011. Generic schema matching, ten years later. In *Proceedings of the VLDB Endowment,* vol. 4, no. 11, pp. 695-701.

Bilke, A. and Naumann, F., 2005. Schema matching using duplicates. In *Proceedings of the 21st International Conference on Data Engineering* (*ICDE*), IEEE Computer Society, Washington, USA, pp. 69-80.

Blake, R., 2007. A Survey of schema matching research. *College of Management Working Papers*, University of Massachusetts Boston, Paper 3.

Census 2014, accessed 3 March 2014,< http://archive.ics.uci.edu/ml/datasets.html >.

Chua, C.E.H., Chiang, R.H. and Lim, E., 2003. Instance-based attribute identification in database integration. *The VLDB Journal,* vol. 12, no. 3, pp. 228-243.

Cilibrasi, R. and Vitanyi, P., 2004. Automatic meaning discovery using Google. *manuscript, CWI.*

Cilibrasi, R.L. and Vitanyi, P.M., 2007. The Google similarity distance. *Journal of Knowledge and Data Engineering*, *IEEE Transactions*, Vol. 19, No. 3, pp. 370-383.

Cortez, E., da Silva, A. S., Gonçalves, M. A., and de Moura, E. S., 2010. Ondux: on-demand unsupervised learning for information extraction. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pp. 807-818.

Dai, B.T., Koudas, N., Srivastava, D., Tung, A., and Venkatasubramanian, S., 2008. Validating multi-column schema matchings by type. *Journal of Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on* IEEE, pp. 120-129.

De Carvalho, M.G., Laender, A.H., Gonçalves, M.A. and Da Silva, A.S., 2013. An evolutionary approach to complex schema matching. *Information Systems,* vol. 38, no. 3, pp. 302-316.

Doan, A. and Halevy, A.Y., 2005. Semantic integration research in the database community: A brief survey. *AI magazine,* vol. 26, no. 1, pp. 83-94.

Doan, A., Domingos, P. and Halevy, A.Y., 2001. Reconciling schemas of disparate data sources: A machine-learning approach. *ACM Sigmod Record* ACM, pp. 509-520.

Doan, A., Domingos, P., and Levy, A.Y., 2000. Learning Source Description for Data Integration. In *Proceedings of the International Workshop on the Web and Databases (WebDB)*, Dallas, USA, pp. 81-86.

Euzenat, J., and Shvaiko, P., 2007. *Ontology matching.* Springer-Verlag, Heidelberg (DE).

Feng, J., Hong, X., and Qu, Y., 2009. An instance-based schema matching method with attributes ranking and classification. In *Proceedings of the 6th International Conference on Fuzzy Systems and knowledge Discovery*, IEEE Press, NJ, USA, Vol. 5, pp. 522-526.

Friedl, J., 2006. *Mastering regular expressions,* O'Reilly Media, Inc.

Hai, D., 2007. *Schema matching and mapping-based data integration: Architecture, approaches and evaluation.* VDM Verlag.

Kang, J., and Naughton, J. F., 2003. On schema matching with opaque column names and data values. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data* ACM, pp. 205-216.

Kang, J., and Naughton, J. F., 2008. Schema matching using interattribute dependencies. *Journal of Knowledge and Data Engineering, IEEE Transactions on,* vol. 20, no. 10, pp. 1393-1407.

Khan, L., Partyka, J., Parveen, P., Thuraisingham, B., and Shekhar, S., 2011. Enhanced Geographically-Typed Semantic Schema Matching. Web Semantics: Science, Services and Agents on the World Wide Web, vol. 9, no. 1, pp. 52-70.

Kleene, S.C., 1951. Representation of events in nerve nets and finite automata. Automata Studies, Princeton University Press, Princeton, NJ, pp. 3-42.

Li, W., and Clifton, C., 1994. Semantic integration in heterogeneous databases using neural networks. *VLDB*, pp. 1-12.

Li, W. and Clifton, C., 2000. SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. *Journal of Data & Knowledge Engineering*, vol. 33, no. 1, pp. 49-84.

Li, W., Clifton, C. and Liu, S., 2000. Database integration using neural networks: Implementation and experiences. *Knowledge and Information Systems,* vol. 2, no. 1, pp. 73-96.

Li, Y., Liu, D., and Zhang, W., 2005. Schema matching using neural network. *Web Intelligence, 2005.* In *Proceedings. The 2005 IEEE/WIC/ACM International Conference on*IEEE, pp. 743-746.

Liang, Y., 2008. An instance-based approach for domain-independent schema matching. In *Proceedings of the 46th Annual Southeast Regional Conference* (*ACM-SE*). ACM, New York, USA, pp. 268-271.

Liu, G., Huang, S. and Cheng, Y., 2012. Research on Semantic Integration across Heterogeneous Data Sources in Grid. In *Frontiers in Computer Education*, Springer Berlin Heidelberg, pp. 397-404.

Mehdi, O.A., Ibrahim, H. and Affendey, L.S., 2012. Instance based Matching using Regular Expression. In *Procedia Computer Science,* vol. 10, pp. 688-695.

Rahm, E., and Bernstein, P.A., 2001. A survey of approaches to automatic schema matching. *The VLDB Journal,* vol. 10, no. 4, pp. 334-350.

Restaurant 2014, accessed 3 March 2014,< http://www.cs.cmu.edu/~mehrbod/RR/>.

Shvaiko, P., and Euzenat, J., 2005. A survey of schema-based matching approaches. In *Journal on Data Semantics IV* Springer, pp. 146-171.

Yang, Y., Chen, M. and Gao, B., 2008. An effective content-based schema matching algorithm. *Future Information Technology and Management Engineering, 2008. FITME'08. International Seminar on* IEEE, pp. 7-11.

.