An Instance based Approach to Find the Types of Correspondence between the Attributes of Heterogeneous Datasets

Muhammad Atif Riaz
Sameer Munir

School of Computing and Engineering
Blekinge Institute of Technology
SE-371 79 Karlskrona
Sweden
This thesis is submitted to the School of Engineering at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science and Mathematical Modeling and Simulation. The thesis is equivalent to 20 weeks of full time studies.

Contact Information:
Authors:
Muhammad Atif Riaz  
E-mail: atifriaz83@gmail.com  
Sameer Munir  
E-mail: samgul.32@hotmail.com

External advisor:  
Patrick Gullin 
Company/Organisation name: Ericsson Telecommunication Karlskrona 
Phone: +46

University advisors:  
Professor Dr. Bengt Carlsson 
School of computing 
E-mail: bengt.carlsson@bth.se 
Associate Professor Dr. Håkan Lennerstad 
School of Engineering 
E-mail: hakan.lennerstad@bth.se

School of Computing and Engineering  
Blekinge Institute of Technology  
SE-371 79 Karlskrona  
Sweden  
Internet : www.bth.se/com 
Phone : +46 455 38 50 00 
Fax : +46 455 38 50 57
ABSTRACT

Context: Determining attribute correspondence is the most important, time consuming and knowledge intensive part during databases integration. It is also used in other data manipulation applications such as data warehousing, data design, semantic web and e-commerce.

Objectives: In this thesis the aim is to investigate how to find the types of correspondence between the attributes of heterogeneous datasets when schema design information of the data sets is unknown.

Methods: A literature review was conducted to extract the knowledge related to the approaches that are used to find the correspondence between the attributes of heterogeneous datasets. Extracted knowledge from the literature review is used in developing an instance based approach for finding types of correspondence between the attributes of heterogeneous datasets when schema design information is unknown. To validate the proposed approach an experiment was conducted in the real environment using the data provided by the Telecom Industry (Ericsson) Karlskrona. Evaluation of the results was carried using the well known and mostly used measures from information retrieval field precision, recall and F-measure.

Results: To find the types of correspondence between the attributes of heterogeneous datasets, good results depend on the ability of the algorithm to avoid the unmatched pairs of rows during the Row Similarity Phase. An evaluation of proposed approach is performed via experiments. We found 96.7% (average of three experiments) F-measure.

Conclusions: The analysis showed that the proposed approach was feasible to be used and it provided users a mean to find the corresponding attributes and the types of correspondence between corresponding attributes, based on the information extracted from the similar pairs of rows from the heterogeneous data sets where their similarity based on the same common primary keys values.

Keywords: Attribute Correspondence, Heterogeneous databases schema matching, Instance based matching.
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSV</td>
<td>Comma separated values</td>
</tr>
<tr>
<td>TFIDF</td>
<td>Term Frequency Inverse Document Frequency</td>
</tr>
<tr>
<td>STFIDF</td>
<td>Soft Term Frequency Inverse Document Frequency</td>
</tr>
<tr>
<td>ILA</td>
<td>Information Learning Agent</td>
</tr>
<tr>
<td>SNM</td>
<td>Sorted Neighborhood Method</td>
</tr>
<tr>
<td>TAILOR</td>
<td>Record Linkage Toolbox</td>
</tr>
<tr>
<td>DCS</td>
<td>Duplicate Count Strategy</td>
</tr>
<tr>
<td>SB</td>
<td>Sorted Block</td>
</tr>
<tr>
<td>HDSM</td>
<td>Heterogeneous Database Schema Matching</td>
</tr>
<tr>
<td>CSM</td>
<td>Complex Semantic Matching</td>
</tr>
<tr>
<td>MUVIS</td>
<td>Multi User View Integration System</td>
</tr>
<tr>
<td>ISI</td>
<td>Intelligent Schema Integration</td>
</tr>
<tr>
<td>SDM</td>
<td>Semantic Data Model</td>
</tr>
<tr>
<td>DUMAS</td>
<td>Duplicate Based Matching of Schema</td>
</tr>
<tr>
<td>WHIRL</td>
<td>Word Based Heterogeneous Information Representation Language</td>
</tr>
</tbody>
</table>
ACKNOWLEDGEMENT

First of all, we would like to thank Almighty Allah who is the most beneficent and merciful.

We are heartily thankful to our academic supervisors Dr. Bengt Carlsson and Dr. Håkan Lennerstad for giving us the opportunity to work under their supervision and providing us with guidelines and suggestions throughout the thesis. They gave detailed feedback and comments which helped us a lot to improve the quality of the thesis document. We appreciate support, technical guidance and encouragement given by our industrial supervisor Patrick Gullin from Ericsson Telecommunication Karlskrona.

We would also like to appreciate the support provided by Dr. Bengt Aspvall and Dr. Lars Lundberg. Finally we would like to thank our families and friends. Their prayers and love was the source of inspiration and relief for us.

Atif and Sameer
# TABLE OF CONTENTS

**Table of Contents**

An Instance based Approach to Find the Types of Correspondence between the Attributes of Heterogeneous Datasets .......................................................................................................................... i

Abstract ................................................................................................................................................... iii

LIST OF ABBRIVATIONS ............................................................................................................................ iv

Acknowledgement .................................................................................................................................... v

TABLE OF CONTENTS ............................................................................................................................... vi

LIST OF FIGURES ...................................................................................................................................... ix

1. INTRODUCTION ............................................................................................................................... 1
   1.1 Overview .................................................................................................................................. 1
   1.2 Problem Statement ................................................................................................................. 2
   1.3 Aims and Objectives ................................................................................................................ 4
   1.4 Motivation and Research Questions ....................................................................................... 4
   1.5 Research Methodology ........................................................................................................... 5
   1.6 Thesis Outline .......................................................................................................................... 5

2. BACKGROUND AND RELATED WORK .............................................................................................. 6
   2.1 Background ............................................................................................................................. 6
   2.2 Related Terms .......................................................................................................................... 7
   2.3 Related Work ............................................................................................................................ 8
      2.3.1 Row/Record Similarity .................................................................................................... 8
      2.3.2 Term/Field Similarity .................................................................................................... 8

3. THEORETICAL WORK ..................................................................................................................... 10
   3.1 Literature Review .................................................................................................................. 10
      3.1.1 Search Strategy .............................................................................................................. 10
      3.1.2 Electronic Database ....................................................................................................... 11
      3.1.3 Study Selection Criteria ................................................................................................. 11
      3.1.4 Conducting Review ........................................................................................................ 12
      3.1.5 Study Selection .............................................................................................................. 13
      3.1.6 The Analysis ................................................................................................................... 14
   3.2 Approaches ............................................................................................................................ 14
      3.2.1 By Comparing Attribute Names ..................................................................................... 14
3.2.2 By comparing attribute values ................................................................. 15
3.2.3 By comparing attribute pattern .............................................................. 16
3.2.4 By comparing field specification ........................................................... 16
3.2.5 Character based field matching ............................................................... 16
3.2.6 Token based field matching ................................................................. 17
3.2.7 Q-Gram Based field matching ............................................................... 18
4 DESIGN AND IMPLEMENTATION OF THE ALGORITHM ...................................................... 19
4.1 Architectural Overview of the Algorithm ................................................. 20
4.2 Algorithm Description ............................................................................ 21
4.2.1 Row Similarity .................................................................................... 22
4.2.2 Term or field Similarity ....................................................................... 24
4.2.3 Attribute Similarity ........................................................................... 25
4.2.4 Types of Correspondence .................................................................... 25
5 EMPIRICAL EVALUATION .................................................................................... 28
5.1 Experiment ......................................................................................... 28
5.1.1 Experimental Environment .................................................................. 28
5.1.2 Experimental test data generator ....................................................... 28
5.1.3 Result evaluation measures .............................................................. 29
5.2 Row Similarity ...................................................................................... 30
5.2.1 Experiment 1 .................................................................................. 30
5.2.2 Experiment 2 .................................................................................. 31
5.3 Term or Field and Attribute Similarity .................................................. 34
5.3.1 Experiment 3 .................................................................................. 34
5.4 Experiment 4 ...................................................................................... 35
6 DISCUSSION ................................................................................................. 37
6.1 Validity Threats .................................................................................... 38
7 CONCLUSIONS ............................................................................................. 40
7.1 Answers to the Research Questions ....................................................... 40
7.2 Future Work ....................................................................................... 40
8 REFERENCES ................................................................................................. 41
9 APPENDIX .................................................................................................... 45
9.1 With no false positives ........................................................................ 45
9.2 With false positive ............................................................................... 45
9.3 After removing false positives .............................................................. 48
9.4 Different number of similar rows used for finding attribute correspondence ...................... 49
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Consistency Checking Procedure</td>
<td>2</td>
</tr>
<tr>
<td>3.1</td>
<td>Multi Step Filtering Process</td>
<td>12</td>
</tr>
<tr>
<td>4.1</td>
<td>System Architectural View</td>
<td>20</td>
</tr>
<tr>
<td>4.2</td>
<td>Sliding Windows</td>
<td>21</td>
</tr>
<tr>
<td>5.1</td>
<td>Manually and Automatic Calculated Correspondence</td>
<td>29</td>
</tr>
<tr>
<td>5.2</td>
<td>Row Similarity Results</td>
<td>31</td>
</tr>
<tr>
<td>5.3</td>
<td>Row Similarity Result with False Positive</td>
<td>33</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Row Similarity Result without False Positive</td>
<td>34</td>
</tr>
<tr>
<td>5.4</td>
<td>Precision Recall and F-Measure</td>
<td>36</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table 1.1: Types of Correspondence between the Attributes ........................................3
Table 3.1: Search Terms ..................................................................................................10
Table 3.2: Database Results after removing duplicates .............................................12
Table 3.3: Selected Studies .........................................................................................14
Table 4.1: Data Set 1 ....................................................................................................22
Table 4.2: Data Set 2 ....................................................................................................22
Table 5.1: Attributes of Data Set 1 && Data Set 2 ......................................................30
Table 5.2: Attributes of Data Set 1 && Data Set 2 ......................................................32
Table 5.3: Term Similarity Results ..............................................................................35
CHAPTER 1

1. INTRODUCTION

1.1 Overview

Telecommunication industries produce and store a huge amount of data which include customer’s profiles, customer call detail data and network data. Telecommunication industry contains billions of records and is amongst the largest in the world [1]. Databases are use to manage and store the information in tabular form. When the size of information increases, the organizations are forced to store information in multiple heterogeneous databases depending upon their current usage scenario [2]. The increased number of data sources requires increased need for control of consistency of the data. Lack of consistent data causes often leakage of revenue. Inconsistent data is a serious problem for businesses leading to incorrect decision making, inefficient daily operations and ultimately wasting both valuable time and money. Having erroneous, duplicate or incomplete data leads to ineffective marketing, operational inefficiencies, inferior customer relationship management, and poor business decisions [3]. Inconsistent data makes it hard for the business to achieve its financial and strategic goals.

Data is thought of as one of an organizations greatest asset but in order to achieve its full potential one requires integrating data from a wide variety of different formats and structures. Potential data sources include legacy systems, enterprise applications, databases, third party data providers and the web. All of these data sources are represented with different names, addresses, phone numbers, email addresses and other data values. Taking the time and effort to pull all this data together without a thought out the data quality process will not only turn out to be costly, but may have an effect on the success or failure of an entire project [4]. Mobile operators have increasing need for consistent data in their network. The number of involved system increases always, and many systems persist its own data which is almost always related to the data in other systems.

The Telecom industry uses consistency checkers to check the consistency of the data collected from arbitrary data sources to understand the problem in the network and to take appropriate action. A consistency checker will make sure that no inconsistent data has been created among the redundant network elements. Data are usually dumped in CSV files that are used as input to the consistency checker. Consistency checker does the comparison of data according to the predefined rules. The purpose of this thesis is to develop an algorithm that take two data sets as input and produce rules that show how attributes from both data sets are related to each other. These rules will be used in the consistency checker to check data consistency as shown in the figure below.
1.2 Problem Statement

We are given two relational datasets with hundreds of millions of rows and hundreds of columns. The problem is to find out the types of correspondence between the attributes of both data sets and the schemas design information of the given datasets is unknown. Correspondence types are equal, not equal, equal ending, and contains in.

The things that are known about the datasets

1. Data sets are in CSV format.
2. Both data sets are in sorted order, i.e. the rows occur in lexicographic order with respect to the common key. Furthermore, it is not known in the start which columns are the common keys.
3. Both datasets share a common potential primary key.

For example we have two rows R and S from dataset one and two respectively that represent the same entity “209398821”.

<table>
<thead>
<tr>
<th>R</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>209398821</td>
<td>Fedrick</td>
<td>Harris</td>
<td>Pholhemsgaton,104</td>
<td>0046762076754</td>
<td>Male</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td></td>
</tr>
<tr>
<td>0762076754</td>
<td>209398821</td>
<td>Male</td>
<td><a href="mailto:fedrick.harris@domain.com">fedrick.harris@domain.com</a></td>
<td></td>
</tr>
</tbody>
</table>
The types of correspondence between the attributes of row S and R are shown in the table below. Row S and R are presented vertically and horizontally respectively in the table.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>“Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>I</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Equal”</td>
</tr>
<tr>
<td>J</td>
<td>“Not Equal”</td>
<td>“J_CONTAIN_B”</td>
<td>“J_CONTAIN_B”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
</tbody>
</table>

**Table1.1: Types of Correspondence b/w Attributes**

The main problem consists of three sub problems first we have to search for the pairs of rows that represent the same real world entities in both datasets called approximate duplicate. Second identifies corresponding attributes by comparing data values within those similar pair of rows. Third determine which types of correspondence are exist between those corresponding attributes.

Finding the pairs of rows that represents the same real world entity from different datasets is very challenging especially when only one or few overlapping attributes are exists. Similarity measure between the rows based on the overlapping attributes, it will be high when numbers of overlapping attributes are high and will decrease as number of overlapping attributes decrease. In our problem both datasets share one common key we called it as potential primary key and only those pair of rows that have the same potential key values are similar. Finding the similar pairs of rows based on the one overlapping attribute is tricky because there will be so many non corresponding attributes values which could coincidently be similar although their respective rows are not representing same real world entity. Schemas design information of both datasets is unknown so it is unclear which attribute in one row to compare with which attribute in the other, this creates problems for finding corresponding attributes which are going to be use for determining correspondence types between attributes of heterogeneous datasets.

The goal of our thesis is to develop an algorithm that utilizes the data available in the data sets to identify attribute correspondence types.
1.3 Aims and Objectives

The main aim of the thesis is to develop an appropriate algorithm for finding correspondence between the attributes of different data sets when schema information is unknown and overlap between the attributes are very low.

Following are the main aims and objectives of the proposed thesis.

- To investigate the current approaches of finding correspondence between the attributes of different datasets.
- To design and implement an algorithm for finding correspondence between the attributes of heterogeneous datasets.
- To validate proposed algorithm on real data provided by telecom industry (Ericson Karlskrona).

1.4 Motivation and Research Questions

In order to achieve the aims and objectives of the study, following research questions were formed.

**RQ1:** What is the current state of art for identifying attribute correspondences in heterogeneous databases?

**Elaboration:** This research question helped to investigate the current approaches and techniques that are used for finding attributes correspondence between heterogeneous databases.

**RQ2:** How do we find types of correspondence between the attributes of different data sets when schema design information of the data sets is unknown?

**Elaboration:** We are focusing on delivering an implementable algorithm that can be used to find out the type of correspondence (equal, not equal, equal ending, contains in) between the attributes of the different data sets. Our proposed algorithm will be generic in sense that it will work on any two data sets. To prove our algorithm’s correctness we will implement our algorithm in java and will conduct experiment with real data sets. To validate the experimental results, we will perform statistical analysis to find out the performance and time complexity of the algorithm.
1.5 Research Methodology

The way of conducting a research or solving the research problem scientifically is called research methodology. Research methodology is a way to systematically solve the research problem and for developing and achieving research goal, it involves the relevant research methods and techniques [6]. The researches strategies are used in this master thesis are literature review, experiment and analysis. Literature review was conducted first to generate underlying theoretical concepts to find the attribute correspondence, challenges and approaches during the integration of heterogeneous databases. The knowledge extracted from literature review was used to develop an algorithm for finding attributes correspondence between different datasets. An experiment was conducted to measure the performance and accuracy of the algorithm.

1.6 Thesis Outline

The documentation of the thesis is organized as follows:

- **Chapter 1**: Consists of Thesis overview, Problem statement, Aims and Objectives, Research questions and Research Methodology.
- **Chapter 2**: Consists of Background, Related Terms and Related work are discussed in this chapter.
- **Chapter 3**: Presents literature review, selected studies and approaches to find attributes correspondence.
- **Chapter 4**: Presents Design and Implementation of the Algorithm, Architectural Overview of the algorithm.
- **Chapter 5**: Discusses Empirical Evaluation of the algorithm, Experiment, Experiment Design and Result of the Experiment.
- **Chapter 6**: Discussion.
- **Chapter 7**: Presents Conclusions and Future Work.
- **Chapter 8**: Presents References.
- **Chapter 9**: Consists of Appendix.
CHAPTER 2

2 BACKGROUND AND RELATED WORK

2.1 Background

Determining attribute correspondence is the most important, time consuming and knowledge intensive part during databases integration. There are different steps in databases integration, that are extracting semantics, transforming formats, identifying attribute correspondence, resolving and modeling heterogeneity, multi-database query processing and data integration [9]. Identifying attribute correspondence more formally called schema matching is the most important step in data integration and is also used in other data manipulation applications such as data warehousing, data design, semantic web and e-commerce. Schema matching is the process of identifying semantically similar attributes in two different databases. In past by using graphical user interface schema matching is typically performed manually was very time consuming, tedious, error-prone and expensive process [44]. It is the most daunting challenges among those who are concerning semantics because implicit knowledge known only by human experts must be made explicit [31]. In literature for schemas matching two approaches are used one schema based matching and other instance based matching [28, 31, 48]. Instance based matcher are much slower and more accurate than the schema based matcher because they rely on actual datasets rather than just schema information [28]. Schema based matcher can able to identify attributes that having same semantics when databases using well standardized names but cannot verify whether they reference the same real objects [28]. In our problem schema information is unknown so we cannot use schema based approach for schema matching. We will use instance based schema matching approach.

In literature Instance based schema matching has been given many names including instance identification, merge or purge, approximate record matching, entity identification, duplicate detection and record linkage [48]. Instance based approaches exploit properties of underlying data for schema matching. In [35] authors proposed an approach for finding corresponding attributes that employs schema and instance information as well as properties of attributes derived from their instances. To find corresponding attributes they performed statistical analysis of the data in duplicates. Identification of duplicates is based on the common ID. The information learning agent (ILA) [49] used instance based approach to learn the schema of other information sources based on the known schema of one source. To identify two pair of objects as duplicates ILA relies on high extensional overlaps. W. Chen [50] proposed an instance based approach to find correspondence between the attributes when column names and data values are difficult to explain or opaque in schemas. His algorithm takes two tables as input and constructs a dependency graph between the attributes by using mutual information, in the later phase graph matching algorithm use to find the matching pair from the graph.

All the approaches described above require the schema under consideration to be aligned and consider two rows are duplicated if overlapping attributes are high between them. The main purpose of the thesis is to find types of correspondence between the attributes by identifying the pair of rows that represent the same real world entity in different datasets where only one attribute is common between those rows. Finding the similar pair of rows based on the one overlapping attribute is tricky because there will be so many non corresponding attributes values could coincidently be similar although their respective rows are not representing same real world entity. Schemas information of both datasets is
unknown so it is unclear which attribute in one row to compare with which attribute in the other creates problems for finding schema matching which is going to use for determining types of correspondence between attributes of heterogeneous datasets.

2.2 Related Terms

- **Attribute**
  One of the fields in a relation is called attribute.

- **Row/Record/Entity Matching**
  The process of matching rows or tuples or records from corresponding relations in heterogeneous databases.

- **Attribute Correspondence/Similarity**
  The process of matching attributes from corresponding relations in heterogeneous databases.

- **Schema**
  A design or representation of plan in the form of an outline or model.

- **Schema Matching**
  The process of matching schema elements from heterogeneous databases.

- **Correspondence Types**
  The relation under which two attributes are corresponding.

- **Row Similarity**
  Row similarity is the process of evaluating the degree of similarity between the pair of rows across different data sets to determine whether they refer to the same real world entity or not.

- **Term or Field Similarity**
  A field in a row is called term. The term similarity is the process of evaluating the degree of similarity between the pairs of terms across the rows that represent same real world entities.
2.3 Related Work

This thesis work is focused on finding attributes correspondence based on combination of two major research topics row similarity and term similarity, due to this related work described under two heading Row Similarity and Term Similarity.

2.3.1 Row/Record Similarity

In the literature various approaches and techniques have been proposed for finding similarity between pairs of rows. A set of record that refers to the same entity can be used in two ways. First it can be use for data cleaning by considering one record as correct and the other record as duplicates containing erroneous information. Secondly both records can be merged into one record with more complete information. TAILOR [34] is a probabilistic record linkage toolbox that serves as a framework for the record linkage, producing a comparison vector by comparing corresponding attributes for each record pair. TAILOR uses linkage rule that assigns each observed comparison vector with a probability of each class to classified row pair as matched, partially matched and unmatched. In [37], the authors proposed an approach called sorted Neighborhood Method (SNM) for record matching when the size of the datasets is very large. The approach employed a window of fixed size v, which moves sequentially over the sorted dataset. The new record enters in the window and compare with the previous v-1 records to find matching records. To increase the chances of falling duplicate into the same window, the sliding window is applied throughout the whole dataset during each single pass. This technique based on the multiple passes with small windows which increase the number of successful matches and decreases the false positives leading to high overall accuracy of the record detection. A disadvantage of this approach is that the window size is fixed. If it selected too small there are chances of missing some duplicates. If it is too large unnecessary comparisons are made. In [46], authors proposed an approach Duplicate Count Strategy (DCS) that uses a varying window size. It is based on the intuition, suggesting large window size for the regions of high similarity and small window size for the regions of low similarity. In [47], the authors proposed an approach called Sorted Blocks. It is a generalization of blocking and windowing. An advantage of Sorted Blocks over SNM is the variable partition size instead of a fixed size window. Variable partition size allows more comparisons if many records are similar and fewer comparisons if few records are similar.

2.3.2 Term/Field Similarity

In [31], the authors have proposed a horizontal approach of attribute matching by exploiting duplicates in the data sets. The approach consists of two steps, duplicate detection and schema matching. For duplicate detection they used cosine similarity with TFIDF well known weighting schema. For schema matching they used normalized edit distance with softTFIDF. In [35], the authors proposed an attribute identification method that employs schema and summary instance information as well as properties of attributes derived from their instances. For identifying similarities between respective attributes they performed statistical analysis of the data in identified duplicate rows. There work is mostly based on textual data.

During integration of heterogeneous databases finding attribute correspondence is a time consuming and critical step. Most of the research on finding attribute correspondence has used correlation analysis techniques, these techniques are only useful for numeric attributes that are linearly related. In [36], the
authors proposed an approach that is applicable on both numeric and character attributes, regardless whether or not they are linearly related. To compare pair of attributes proposed approach considers both matching and non matching pair of records from the databases applies matching function and then to measure the dependency of pair of attributes and class of the pair of records it uses mutual information. The proposed procedure consists of three steps:

1. A sampled set of matching and non matching pairs of records from heterogeneous databases.
2. Similarity measure of the attributes values for each record pair in the sample by using exact or approximated matching function.
3. Compute the mutual information between the class of record pairs and the attribute matching function value for each attribute pair.

W. Chen, H. Guo, F. Zhang [44] proposed a new algorithm HDSM used to calculate corresponding semantic relations between the attributes of heterogeneous databases during the process of data integration. HDSM consists of five steps preprocessor, match generator, similarity estimator, match selector and supplemental match generator. This algorithm can not only discover 1:1 matching but also discover complex matching such as 1:n, n:1 and m:n. HDSM can also discover matching between opaque column names and data values.

To find semantic correspondence between the attributes of two different schemas Y. Qian, Y. Li, J. Song, and L. Yue [45] introduced an approach called CSM. In this approach the steps preprocessor and clustering processor using data types and values to filter unreasonable matches. To discover 1:1 and complex matches in match generator special purpose searchers are used to explore a specialized portion of the search space. At the end by using similarity estimator to estimate candidate matches and match selector to selects optimal candidate matches.
CHAPTER 3

3 THEORETICAL WORK

3.1 Literature Review

Literature Review is an initial and most important step for any research to provide the researcher with an up to date account and discussion of the research findings in a particular topic. There are different reasons for undertaking the literature review are [7][8].

- Including identification of gaps in the current research.
- To carry on from where others have already reached.
- To identify information and methods that is relevant to our research area.
- To find the methods and techniques that is used for finding the correspondence between the attributes of the heterogeneous databases.

3.1.1 Search Strategy

In order to generate search strategy following steps were followed.

3.1.1.1 Construction of Search Terms

In order to construct the search terms used for the study the following steps were followed.

- Develop the search terms from research questions.
- All search terms were listed that are related to research questions.
- Synonyms of all search terms were included.
- New search terms were added by changing plurals search terms to singular and singular to plurals.
- Boolean operator OR were used with synonyms of the search terms.
- Boolean operator AND were used with antonyms of the search terms.

All search terms are listed in the table below

<table>
<thead>
<tr>
<th>No.</th>
<th>Search Terms</th>
<th>No.</th>
<th>Search Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Correspondence</td>
<td>9</td>
<td>Method</td>
</tr>
<tr>
<td>2</td>
<td>Relationship</td>
<td>10</td>
<td>Strategy</td>
</tr>
<tr>
<td>3</td>
<td>Attribute</td>
<td>11</td>
<td>Process</td>
</tr>
<tr>
<td>4</td>
<td>Approach</td>
<td>12</td>
<td>Framework</td>
</tr>
<tr>
<td>5</td>
<td>Technique</td>
<td>13</td>
<td>Heterogeneous datasets</td>
</tr>
<tr>
<td>6</td>
<td>Heterogeneous Databases</td>
<td>14</td>
<td>Schema Matching</td>
</tr>
<tr>
<td>7</td>
<td>Different databases</td>
<td>15</td>
<td>Different datasets</td>
</tr>
<tr>
<td>8</td>
<td>Instance based schema</td>
<td>16</td>
<td>Instance based schema</td>
</tr>
</tbody>
</table>

Table 3.1: Search Terms
3.1.2 **Electronic Database**

Different database sources that are used for searching the relevant literature are listed below.

- IEEE Explore
- ACM Digital Library
- Science Direct
- Springer Link
- Engineering Village (Inspec - Compendex)
- ISI web of knowledge

Our research was not limited within the above mentioned databases only, we used various other sources including journals and white papers published by companies, technical research reports and other online resources found on the internet were used. Kitchenham[7] described research strategies were followed. By using search terms and Boolean operator search strings were created. These search strings were run on Metadata search and where Metadata search were not available search strings were run on Abstract, Title and Keywords in the electronic research databases.

3.1.3 **Study Selection Criteria**

After electronic databases sources were selected, studies selection criteria inclusion and exclusion were defined. Inclusion criterion describes the rules by which relevant articles are included in the study and exclusion criterion describes the rules by which irrelevant articles are excluded from the study.

3.1.3.1 **Inclusion Criteria**

- The language of the article should be English.
- The abstract and title should match the problem domain.
- The complete article should match the problem domain.
- The article should be available in full text.
- The article should be peer reviewed.
- Remove the papers that have been duplicated, as same articles were found on more than one database.

3.1.3.2 **Exclusion Criteria**

- The studies which do not related to attributes correspondence.
- The studies which do not related to approaches for finding attributes correspondence.
- If full text of the study was not available.
- Exclude the studies which are not in English language.

Above mentioned criteria were applied in three different levels
### 3.1.4 Conducting Review

In this phase article was extracted from the electronic databases and selected after implanting inclusion, exclusion criterion. The collected articles were critically reviewed by reading abstract, introduction and conclusion of the articles, only the 34 articles relevant and useful for our problem domain were included by finally going through the whole text. The conducting review is shown in the figure 3.1 below.

![Figure 3.1: Multistep filtering process](image)

The numbers of studies found in different electronic databases after removing duplicates are shown in the table 3.2.

<table>
<thead>
<tr>
<th>Databases</th>
<th>Total Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACM Digital Library</td>
<td>378</td>
</tr>
<tr>
<td>IEEE Xplorer</td>
<td>643</td>
</tr>
<tr>
<td>Springer Link</td>
<td>262</td>
</tr>
<tr>
<td>Science Direct</td>
<td>197</td>
</tr>
<tr>
<td>Engineering Village</td>
<td>358</td>
</tr>
<tr>
<td>ISI Web of Science</td>
<td>137</td>
</tr>
<tr>
<td><strong>Total Studies Found</strong></td>
<td><strong>1975</strong></td>
</tr>
</tbody>
</table>

**Table 3.2:** Database Results after removing duplicates
### Study Selection

Table 3.3 shows the peer reviewed studies which were selected as a result of performing literature review.

<table>
<thead>
<tr>
<th>Sr.no</th>
<th>Title</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>So far (schematically) yet so near (semantically) [10]</td>
<td>A. Sheth and V. Kashyap</td>
</tr>
<tr>
<td>3</td>
<td>Multi-user view integration system (MUVIS): an expert system for view integration [12]</td>
<td>Hayne S. and Ram S.</td>
</tr>
<tr>
<td>5</td>
<td>SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks [9]</td>
<td>Li Wen-Syan and Clifton Chris</td>
</tr>
<tr>
<td>8</td>
<td>Improving database quality through eliminating duplicate records [23]</td>
<td>M. Wei, A. H. Sung, and M. E. Cather</td>
</tr>
<tr>
<td>11</td>
<td>Federated database systems for managing distributed, heterogeneous, and autonomous databases [18]</td>
<td>A. P. Sheth and J. A. Larson</td>
</tr>
<tr>
<td>13</td>
<td>Schema matching using duplicates [31]</td>
<td>A. Bilke and F. Naumann</td>
</tr>
<tr>
<td>15</td>
<td>Using field specifications to determine attribute equivalence in heterogeneous databases [22]</td>
<td>W.-S. Li and C. Clifton</td>
</tr>
<tr>
<td>16</td>
<td>Binary codes capable of correcting deletions, insertions, and reversals [24]</td>
<td>V. I. Levenshtein</td>
</tr>
<tr>
<td>17</td>
<td>Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida [26]</td>
<td>M. A. Jaro</td>
</tr>
<tr>
<td>19</td>
<td>String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage [27]</td>
<td>W. E. Winkler</td>
</tr>
<tr>
<td>20</td>
<td>Two supervised learning approaches for name disambiguation in author citations [29]</td>
<td>H. Han, L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis</td>
</tr>
<tr>
<td>22</td>
<td>Integration of heterogeneous databases without common domains using queries based on textual similarity [21]</td>
<td>W. W. Cohen</td>
</tr>
</tbody>
</table>
Table 3.3: Selected Studies

<table>
<thead>
<tr>
<th></th>
<th>Study Description</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>A generalization of blocking and windowing algorithms for duplicate detection [47]</td>
<td>U. Draisbach and F. Naumann</td>
</tr>
<tr>
<td>25</td>
<td>Combining schema and instance information for integrating heterogeneous data sources [48]</td>
<td>H. Zhao and S. Ram</td>
</tr>
<tr>
<td>26</td>
<td>Discovering Semantic Matches between Opaque Database Schemas [50]</td>
<td>W. Chen</td>
</tr>
<tr>
<td>28</td>
<td>Discovering Complex Semantic Matches between Database Schemas [45]</td>
<td>Y. Qian, Y. Li, J. Song, and L. Yue</td>
</tr>
<tr>
<td>29</td>
<td>Mining schema matching between heterogeneous databases [44]</td>
<td>W. Chen, H. Guo, F. Zhang, X. Pu, and X. Liu</td>
</tr>
<tr>
<td>30</td>
<td>A survey of approaches to automatic schema matching [40]</td>
<td>E. Rahm and P. A. Bernstein</td>
</tr>
<tr>
<td>31</td>
<td>Exploring Attribute Correspondences Across Heterogeneous Databases by Mutual Information [36]</td>
<td>H. Zhao and E. S. Soofi</td>
</tr>
<tr>
<td>34</td>
<td>Schema matching using duplicates [31]</td>
<td>A. Bilke and F. Naumann</td>
</tr>
</tbody>
</table>

3.1.6 The Analysis

The result obtained from the literature review are analyzed in order to answer research question one. The analysis mainly focuses on the approaches that are used in literature for finding correspondence between attributes of heterogeneous databases. The analysis result is presented in the next section 3.2 under the heading approaches. The knowledge extracted during literature review are used in developing algorithm for finding correspondence types between the attributes of heterogeneous databases as presented in chapter 4.

3.2 Approaches

In the literature various approaches and techniques have been proposed for detecting correspondence between the attributes of heterogeneous databases are listed below.

3.2.1 By Comparing Attribute Names

By comparing attribute name used the concept that the same attribute may be represented by synonyms in different databases. To measure the similarity between the names of the attributes different linguistic techniques such as dictionary, case grammar, conceptual graph are used. A. Sheth and V. Kashyap [10] proposed heuristics to find semantic similarity between objects that have various types of schematic differences and data inconsistency. They also introduced the concept of semantic
proximity to specify degrees of semantic similarity among objects based on their real world semantics. To overcome the semantic differences among objects, they developed taxonomy of schematic conflict.

C. Yu, W. Sun, S. Dao, and D. Keirsey [11] discussed that the semantics of a majority of attributes can be determined by consulting pre-established concepts. Then the whole attributes semantics can be determined by retrieving information from the data dictionary as data dictionary are used to store description of each attribute associated with each concept in a hierarchy, they store its name, name of its dependent concepts and their synonyms. A knowledge-based system Multi User View Integration System (MUVIS) [12] also used to attribute names to find correspondence attributes. MUVIS is knowledge-based system represent user views and integrates these views into a global conceptual view MUVIS is based on object-oriented data model (SDM). Due to use of virtual relationships in SDM reduced the complexity of the integration. By providing heuristic MUVIS procedure was enhanced. By comparing the field name MUVIS determine the similarity or dissimilarity of two objects.

Existing systems that are using attribute names for finding correspondence between the attributes are work by using synonym lexicon such as WorldNet [13]. The problem with this method is when database designers use abbreviations such as NID for national identification number, W Phone for work phone and SSN for social security number, it is almost impossible to find the correspondence by consulting synonym lexicon and homonyms is also another problem [9].

In [5] authors proposed a new tool called SIS (intelligent schema integrator) used to identify the existence of homonym and synonym conflict in the source schema and removed it during the construction of the global schema. To tool provides solution that allows user to identify which elements that they want to maintain or remove by using thesaurus.

By using attribute name of finding correspondence between the attribute of different data sets working well for homonyms but not working well in case of synonym. Synonym lexicon has limitation, it is difficult for database designer to define field name by using only the words that can be found in a dictionary.

### 3.2.2 By Comparing Attribute Values

This approach uses attribute values for determining attribute correspondence. In [15], the authors proposed two approaches. The first approach is based on the mutual information and entropy these feature are extracted from the value of compared attributes by using N-grams technique. This approach produces good result when finding similarity between the attributes of two data sets where few shared instances exist. In the second approach they proposed for semantic similarity between two data sets is based on K-medoid clustering of the keywords extracted from the compared attributes. Normalized Google Distance matrix is used for assigning keywords to clusters.

In [23] the authors proposed a framework treating attribute values as string to resolve the problem by identifying semantically equivalent string values in syntactically different representation. They used a token-based solution and proposed a general frame work for filed matching. They introduced a concept of string matching points to improve string matching accuracy and efficiency.

The approach that is discussed in [16] are using domain relationship i.e Equal, Contains, Overlaps etc to integrating entity sets. The problem with these techniques are finding relationships is a time consuming process and small amounts of incorrect data lead easily to wrong conclusion[9, 17, 18]
another problem with this approach is poor tolerance of faults. To develop an automatic system for
determining the correspondence by using attribute values need to depend on heuristic [17, 19].

The approach DUMAS (duplicate based matching of schema) [31] used instance based matching
algorithm for finding correspondence between the attributes of heterogeneous databases. Each instance
attributes values are compared to draw correlation between them. To generate similarity matrix
between the attributes they used normalized edit distance to calculate similarity between the attributes.
Soft TFDIF was also used to compare attributes that are semantically same but having different
structure.

3.2.3 By Comparing Attribute Pattern

In [20] authors proposed different approaches for finding attribute correspondence by using data
pattern instead of using data values and domains. This technique consumes comparative less time than
the technique using attribute values. To determine similarity between attributes they used a self
organizing classifier algorithm to categorize attributes and then use a back propagation learning
algorithm to train a network to recognize input patterns. The situation where same input pattern have
two different expected outputs the classifier cannot be trained with such kind of training set data.

3.2.4 By Comparing Field Specification

Using field specification for finding attribute correspondence also has been proposed. To determine
correspondence between the attributes, W.S. Li and C. Clifton [22] proposed a technique that is based
on field specifications. This approach provides solution for time consuming procedure of finding
attribute relations [18] by using field specification (schema information). During the process of finding
attribute correspondences this approach can also be used to eliminate those attribute that are
incompatible by comparing attribute domain. Using field specification such as length, data types and
constraints for finding attributes correspondence is an easy approach but thing that make it
computationally expensive is the first step to eliminate clearly incompatible attributes. Other problem
with this approach is that schema information may not be always available.

3.2.5 Character Based Field Matching

Character based string matching algorithms compares two strings i.e. \( A = c_1, c_2, c_3, \ldots, c_n \), \( B =
c_1, c_2, c_3, \ldots, c_m \) character by character. Simple edit distance algorithm is most commonly applied
algorithm in this category is proposed by V I Levenshtein [24], which calculate atomic editing
operations insertions, deletions, transpositions and substitutions that are required to transform one
string to the other. The simplest versions of the algorithm having cast one for each edit operations.
Computing edit distance by using simple version of the algorithm between two string \( a \) and \( b \) of length
\(|a|\) and \(|b|\) takes time \( O(|a|*|b|) \). A two way algorithm proposed by M. Crochemore and D. Perrin
algorithm [25] is linear in time and uses constant space. This algorithm can be viewed as an
intermediate between the algorithm “Knuth, Morris and Pratt” and “Boyer Morre”. There are other
character based field matching algorithms such as M. A. Jaro algorithm [26] and W. E. Winkler algorithm [27].

### 3.2.6 Token Based Field Matching

Using token based field matching algorithms, the problems that have been identified by H. Han et al. [29] and G. Navarro et al. [30] like names abbreviation, identical names, names misspellings, pseudonyms in publication, ordering, word omission and insertion can be solved. Token based algorithms divide string into tokens or set of words like $S = \{w_1, w_2, w_3, ... w_n\}$ and then perform operations on these words for field matching. According to M. Wei et al. [23], there are two obvious reasons that improve string comparison by performing at word level,

1. By using token or word based method, stop words and punctuation can be easily removed from strings and only meaningful strings are left to compare.
2. By using token or word based method, abbreviations can be taken care by expressing words as sequences of characters.

Jaccard similarity matrices is the simplest token based field matching algorithm, which determined the similarity of two strings as,

\[
\text{Ratio of } \frac{N_c}{N_d}
\]

Where $N_c$ total number of common words and $N_d$ total number of distinct words in two strings. It uses word sets from the comparison strings to evaluate similarity. Each data string is represented as a jaccard vector similarity function.

In [23] the authors proposed a frame work treating attribute values as string to resolve the problem by identifying semantically equivalent string values in syntactically different representation. They used token based solution and proposed a general frame work for filed matching. They introduced a concept of string matching points to improve string matching accuracy and efficiency.

In [21] authors proposed a system called WHIRL, to calculate similarity of two attributes WHIRL used cosine similarity with the TF.IDF weighting scheme. WHIRL convert string into tokens and each token is assigned a weight

\[
w(a) = \log(tf_a + 1) \cdot \log(idf_a)
\]

Where $a$ is token in the string $tf_a$ is the term frequency of $a$ and $idf_a$ is the inverse document frequency. Then the similarity between two token $a$ and $b$ will be

\[
sim(a, b) = \frac{\sum_{i=1}^{\|D\|} w_a(i) \cdot w_b(i)}{\|w_a\|_2 \cdot \|w_b\|_2}
\]

This similarity matrix working well with the normal string but when two strings like “Department of Computer Science” and “Computer Science Department” are similar but this similarity matrix will consider as two different strings.
3.2.7 Q-Gram Based Field Matching

This method takes a string and an integer $q$, the set of $q$ gram of $s$ denoted as $G_q(s)$ is obtained by sliding a window of length $q$ over the characters of strings [23]. For example, we have a string “blekinge” and 3 gram set of this string can be represented as

$$G_q("blekinge") = \{\text{'ble'}, \text{'lek'}, \text{'eki'}, \text{'kin'}, \text{'ing'}, \text{'nge'}\}.$$

The similarity measure of two strings can be determined as the ratio of common $q$ grams to the total number of distinct $q$ grams in two strings. H. Köhler, X. Zhou, S. Sadiq, Y. Shu, and K. Taylor [28] proposed new sets of similarity measure between sets of strings using $q$ grams method when integrating data from different sources.
CHAPTER 4

4 DESIGN AND IMPLEMENTATION OF THE ALGORITHM

The knowledge extracted during literature review is used in developing algorithm to find correspondence types between the attributes of heterogeneous datasets. There are two main techniques to find corresponding attributes schema level and instance level [28, 31, 48]. Schema level approach needs schema design information to determine corresponding attributes. Since schema design information was unknown in our case so we used instance level approach. Instance level approach is divided into two types: vertical matching and horizontal matching [31, 48]. In the vertical approach, the characteristics of attributes are compared to find correspondence. Attributes characteristics are extracted by evaluating distribution of characters, string length, order of characters in a string etc. Attributes having similar characteristics are matched. In horizontal matching, datasets are searched for similar rows (records that represent the same real world entities). When similar rows have been identified their terms are compared to find corresponding attributes.

The proposed algorithm used horizontal matching because vertical matching is more tedious and time consuming [9]. Each pair of row produced a term or field similarity matrix. To find attributes correspondence, each term similarity matrices are combined by taking average, the resultant matrix is called attribute similarity matrix. Attribute similarity matrix used to determine the types of correspondence between the corresponding attributes of heterogeneous datasets. Attribute similarity matrix representing corresponding attributes in the form of double numeric values lies between 0 and 1. It is a time consuming process to evaluate each pairs of attributes for each types of correspondence. We defined a heuristic value for each types of correspondence only those attributes will be evaluated for specific correspondence types if their values lie within the heuristic values of that specific type.

For row similarity we have used token based approach called cosine similarity [33] and for term similarity we have used character based approach called Levenshtein edit distance [24, 32].
4.1 Architectural Overview of the Algorithm

The architectural overview of the proposed algorithm is shown in the figure below.

![System Architectural View](image)

**Row Similarity Phase:**
- **Input:** Hundred rows from both data sets in each iteration.
- **Output:** Pairs of correspondence rows.
  1. Compute vector patterns of all record pairs of data set 1 and 2.
  2. Compute normalized weight of each term in the each rows of both data sets.
  3. Calculate cosine similarity between each pairs of rows.
  4. Keep all similar pairs of rows in the candidate set from each iterations.
  5. Removing false positives from candidate set by using known correspondence and resultant group consists of only those pairs of rows that are similar based on same primary keys values.
  6. Send similar pairs of rows one by one to the next phase.

**Term Similarity Phase:**
- **Input:** pair of rows one by one
- **Output:** Combined Similarity Matrix of each iteration.
  1. Compute normalized edit distance between each terms of the similar pair of rows.
  2. Calculate term similarity matrix of each row pair.
  3. Continue 1 and 2 for each pair of rows.
  4. $\frac{1}{n}\sum (all\ the\ term\ similarity\ matrices\ that\ are\ obtained\ in\ current\ iteration)$
  5. $P$ is the number of correspondence rows

**Attribute Similarity Phase:**
- **Input:** Term Similarity matrices of all iterations.
- **Output:** Attribute Similarity Matrix
  1. $\frac{1}{k}\sum (all\ the\ term\ similarity\ matrices\ that\ are\ obtained\ in\ all\ iteration)$
  2. $K$ is the number of iteration.

**Types of correspondence Phase:**
- **Input:** Attribute Similarity Matrix
- **Output:** correspondence types (equal, not equal, contains in, equal ending)
  1. Produce output by comparing attribute similarity matrix with user defined heuristics.

**Figure 4.1:** System Architectural View
4.2 Algorithm Description

**INPUT:** Two Relational data sets

**OUTPUT:** Rules, which type of correspondence between the attribute of the both data sets are having i.e. equal, not equal, equal ending, contain in.

**Prerequisite of the algorithm:** Both data sets should be in CSV format and having records in sorted order.

The given datasets consists of hundreds of millions of rows and each row consists of hundreds of attributes. As we have limited main memory it is practically impossible to bring all data from both data sets into the main memory to perform operations. To deal with this problem, the proposed algorithm will work iteratively on sliding window concept [23, 37, 46]. We are using two sliding windows of flexible size, one for first dataset and other for second dataset as shown in the figure below. The reason behind using flexible size window because fixed size window have two disadvantages [46]. First if the selected window size is too small, similar rows might be missed and if it is selected too large results in many unnecessary comparisons. We have sorted datasets so sliding window will move through the list of sequential rows until it reached the end. Initially both windows size is fixed to 100 rows. The reason behind considering the sample of only 100 rows of each iteration is as we have sorted data sets so related record of an entity from one data set is guaranteed to be there in the other data set with same order if it is not missing. All rows from first window will be compared (using row similarity) with all rows to the other window in search of matching pairs. Suppose we found a last matched pair in the first sample having 80th row from 1st dataset and 70th row from 2nd dataset. In next iteration 1st sliding window will keep the last 20 rows and bring 80 more rows from the 1st dataset and 2nd sliding window will keep the last 30 rows and will bring 70 more rows form the 2nd dataset. In case when no record found similar with in size 100, sliding windows will add 100 more rows from each datasets in this case window size will become 200 rows and same procedure repeat again as described above. The iterations will continue until all the rows from both data sets are not processed.

![Figure 4.2: Sliding Windows](image-url)
The proposed algorithm has four main parts

i) Row similarity
ii) Term similarity
iii) Attribute similarity
iv) Types of correspondence

4.2.1 Row Similarity

Row similarity is the process of evaluating the degree of similarity between the pair of rows across different data sets to determine whether they refer to the same real world entity or not. The higher similarity between the two rows, the higher the probability that they actually refer to the same real world entity. As we have millions of rows in both data sets, comparing each row from the first data set with each row to the other data set especially at term level is infeasible. Also we have relational data sets to compare only rows that have at least one term in common [31]. If we are comparing a pair of row that does not represent the same entity we will not be able to find the exact correspondence between the attributes as shown in the example below.

We have sorted data sets so we can compare directly term similarity between the rows of both data sets, but problems occur when records are missing as shown in the tables above. The row 1 in table 1 cannot be compared directly with the row 1 in table 2 at term level because the row 1 from data set 1 having similarity with row 2 in data set 2 based on the same primary key values (row 1 from data sets 1 and row 2 from data set 2 represent the same entity 1010). Therefore finding term similarity between two rows without finding row similarity is inappropriate, time consuming and this will lead towards wrong conclusion. Row similarity makes sure that pair of rows represents the same real world entity.

The rows in the data sets consists of many token separated by comma to measure row similarity we used token based distance method called cosine similarity or TFIDF [33] i.e. the product of two normalized vectors which is equal to the cosine of the angle between the vectors. There are two reason of using TFIDF similarity measure. First, it is widely used in information retrieval community and comparatively performs better than the other methods [32]. Second, TFIDF measure is order independent [31] as we have unaligned data sets suit our situation.

To measure cosine similarity each row $r_i$ is represented as a vector $r_i = [t_1, t_2, t_3, ..., t_n]$ where $n$ is the number of terms and $t_n$ is represented the $n^{th}$ term and assigned weights to each value $t_i$ by using TFIDF weighting scheme [33]. The row similarity aims at computing the overall similarity $\text{RowSim}(r_1, r_2)$ of $r_1$ and $r_2$, in order to determine whether $r_1$ and $r_2$ refers to same real world entity. TFIDF define weights $w(r, t_i)$ of each term $t_i$ in the row $r$ as
\[ w(r, t_i) = \log(t_{f,r} + 1) \cdot \log(\frac{N}{d_{f,t_i}} + 1) \] [32].

Where \( t_{f,r} \) is the term frequency (TF) of \( t \) in row \( r \), \( N \) is the total number of rows and \( d_{f,t_i} \) is the number of rows in which term \( t_i \) appears. \( \frac{N}{d_{f,t_i}} \) is the inverse document frequency (IDF). The purpose of using IDF is to avoid ambiguity that may occur during row similarity. For example when a term appear frequently in most of rows like a field gender in a data set, value M/F will repeat again and again in each rows, if this field present in both data sets the result would be many rows having at least one term common, hence they can be comparable. The IDF of a term that occur frequently will be low so it will not effect on similarity score. The weight of the such terms will be low as compare with the other terms in the vector that are not frequently appear in data sets.

In both data sets we are treating each row as a single vector and there are different numbers of term in each vector. There may be possibility that one vector has very large magnitude as compare to the other vector. For example vector one from the file 1 having 150 terms and second vector from file 2 have only 15 terms. Due to large number of terms in first vector the Euclidean distance of first vector and second vector is very large, it is insignificant to show their distance in graphs. Normalization of vectors is used to overcome this effect which turn vector into unit length and make their product and representation in the graph easy and meaningful. Every term weight in a vector is divided by the Euclidean length of the TD x IDF weighted vector. Where Euclidian length is given as follows

\[ |v| = \sqrt{w_1^2 + w_2^2 + w_3^2 \ldots w_n^2} \]

Where \( w_i \) is the TF x IDF weight of the \( i^{th} \) term in the vector \( r \). The normalized weight of a term is represented by \( \omega \) and is calculated as

\[ \omega_i = \frac{w_i}{|v|} \] Where \( i \) is from 1 to \( n \).

We produce \( \omega' \) from \( \omega \) by keeping the largest entries in \( \omega \) and replacing all others by zero.

Let’s suppose we have one row \( r_1 \) from data set1 and second row \( r_2 \) is from data set2, the row similarity of \( r_1 \) and \( r_2 \) is calculated as,

\[ \text{RowSim}(r_1, r_2) = \sum_{t \in r_1 \cap r_2} \omega'(r_1, i(t)) \omega'(r_2, i(t)) \] [32].

Where \( \omega' \) is the normalized weight of the common term \( t \) in \( r_1 \) and \( r_2 \) and \( i(t) \) is the index of term \( t \) in vector \( r_1 \).

The algorithm considers similarity between the pairs of rows only when the common attributes of these rows have maximum normalized weights. All similar pairs of rows that are selected are named as group of candidate pairs. In some cases it is possible that our algorithm selects a pair of rows as similar even when similarity measure is not based on the potential primary key. For example we have two rows R and S, R from a sample of first dataset and S from the sample of second dataset.

<table>
<thead>
<tr>
<th>R</th>
<th>atif</th>
<th>House no. 785</th>
<th>00467285154</th>
<th>male</th>
<th>French</th>
</tr>
</thead>
<tbody>
<tr>
<td>1010</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| S       | French | 07562489     | atif        | male |
|---------|--------|---------------|-------------|------|-------------|
Suppose we have sample size 50 rows, the attributes male and French appeared in the sample frequently and they having low IDF as a result the normalized weights of these attributes will be low therefore our algorithm will not consider these attributes in row similarity measure but attribute with value “atif” appeared only once in the sample of both data sets, so the normalized weight of these attributes will be equal to the weight of the primary key that are 1010 and 1001 in both rows. Such rows will also be selected.

To overcome this problem we can use previous knowledge about the attributes correspondence to improve row similarity measure as suggested by Alexander Bilke and Felix Naumann [31]. One thing that is known about the attributes correspondence is that both datasets having one attribute common. We called it potential primary key in both datasets. We used potential key in row similarity measure to correctly identify the pair of rows that are representing same real world entity. We kept record of common terms of the all selected pairs of rows. After finding all similar pairs of rows we considered only those pairs of rows for term similarity which are selected based on the common attributes that are appeared in most pairs of rows.

We introduced a method to find the potential primary keys in both datasets. Consider two datasets consist of sets of rows R and S respectively. Where 
\[ R = \{ r_1, r_2, r_3, ..., r_m \} \]  
\[ S = \{ s_1, s_2, s_3, ..., s_n \} \]
Each row consist of number of terms 
\[ r_i = \{ a_1, a_2, a_3, ..., a_m \} \]  
\[ s_j = \{ b_1, b_2, b_3, ..., b_n \} \]. Let A be a set having all the pair of rows selected as similar including false positive along with a set B having information about the position of the common attributes of selected pair of rows. We define A and B in the following way

\[ A = \{ (x_i, y_j) \mid x_i \in r_i \& y_j \in s_j ; i = 1, 2, ..., m \& j = 1, 2, ..., n \} \]

\[ B = \{ (x_i, y_j) \mid x_i \in a_i \& y_j \in b_j ; i = 1, 2, ..., m \& j = 1, 2, ..., n \} \]

Then we define a set C which gives us an order pair of positions of the common attributes which appears maximum times in B which is represented by \( \lambda \) and is given by

\[ C = \{ (x_i, y_j) = \lambda \mid \lambda \in B \} \]

Where B is the information about the position of the common attributes.

Finally we define a set D which actually gives us exact matching pair of rows after reducing the false positives and is given by

\[ D = \{ (x_i, y_j) \mid \forall B \approx \lambda \Rightarrow (x_i, y_j) \in A; x_i \in r_i \& y_j \in s_j ; i = 1, 2, ..., m \& j = 1, 2, ..., n \} \]

By using above formula the frequency of false positives are reduced as illustrated by the experiment in Section 5.2.2.

4.2.2 Term or field Similarity

A field in a row is called a term. The term similarity is the process of evaluating the degree of similarity between the pairs of terms across the rows that represent same real world entities. Term similarity matrix is calculated for each pairs of rows from the group of candidate pairs one by one.

Consider a pair of rows \( r_1 \) with terms \( t_{a_1}, t_{a_2}, ..., t_{a_m} \) and \( s_1 \) with terms \( t_{b_1}, t_{b_2}, ..., t_{b_n} \), the term
similarity matrix is a \( m \times n \) matrix which gives the term similarity of the terms \( t_{a_i} \) and \( t_{b_j} \). The similarity between each pair of the terms \( t_{a_i} \) and \( t_{b_j} \) is defined as

\[
\text{termsim}(t_{a_i}, t_{b_j}) = 1 - \frac{ed(t_{a_i}, t_{b_j})}{\max\{|t_{a_i}|, |t_{b_j}|\}}; \quad t_{a_i} \in r_1 \quad \text{and} \quad t_{b_j} \in s_1[38]
\]

\[i = 1, 2, \ldots, m \quad \text{and} \quad j = 1, 2, \ldots, n\]

where \( \max\{|t_{a_i}|, |t_{b_j}|\} \) is the length of the longer term and \( ed(t_{a_i}, t_{b_j}) \) is the edit distance between the terms \( t_{a_i} \) and \( t_{b_j} \) which is calculated as the distance between two strings and is defined as the minimum number of edit operations (insertion, deletion, substitution) required to transfer one string into another. Algorithm performs one edit operation at a time with the same uniform cost for each of these operations and then normalized the edit distance calculated by dividing \( ed(t_{a_i}, t_{b_j}) \) with the maximum length of the strings \( t_{a_i} \) and \( t_{b_j} \) [38]. According to the Cho et al. [32], better result has been obtained by using Jaro distance matrix during term similarity. The reason behind using edit distance is, as Jaro matrix performs well for strings with slight spelling variations but does not cope well with longer strings separating common characters due to the restriction that common characters must have to occur in a certain distance from each other [39] and also Jaro distance matrix having higher computation cost than edit distance. Similarly, term similarity of each pair of rows will be calculated. And at the end all the similarity matrices of each pair of rows that are selected for one iteration (as four in our case) are combined into one similarity matrix by using aggregation.

### 4.2.3 Attribute Similarity

All the term similarity matrices that are produced in term similarity phase are combined into one similarity matrix we called it attribute similarity matrix of both data sets \( D_1 \) and \( D_2 \) by using this formula.

\[
\text{AttrSim}(D_1, D_2) = \frac{1}{n} \sum_{i=1}^{n} \text{termsim}(t_{a_i}, t_{b_j})
\]

Where \( n \) is the number of iteration and \( \text{termsim}(t_{a_i}, t_{b_j}) \) represents similarity matrix of each iteration.

### 4.2.4 Types of Correspondence

The attribute similarity matrix represents the correspondence between the attributes of two heterogeneous datasets (attributes of first dataset vertically and attributes of the second dataset horizontally) in the form of double values between 0 and 1. By using these values we are going to define the heuristic for finding the types of the correspondence which are equal, not equal, equal ending and contain in. To find correspondence types between the attributes, heuristic values will help algorithm to explore only those attributes whose similarity values lies between the heuristic intervals.
4.2.4.1 Equal

In the field similarity matrix let x be the value representing the correspondence between the two attributes, if x=1 then the type of correspondence between the attributes will be “equal”.

4.2.4.2 Not Equal

In the field similarity matrix let x be the value representing the correspondence between the two attributes, if $0.00 \leq x \leq 0.009$ then the type of correspondence between the attributes will be “not equal”. The reason behind selecting this interval is as we are using string operation to find the correspondence between attributes there might be some attributes that are not equal but some of their character having similarity due to which similarity matrix will show values between 0.00 to .009.

4.2.4.3 Equal Ending

In our case there is a possibility that an attribute “phone number” present in first dataset with country code and the same attribute present in the other dataset without country code. The correspondence type between these attributes is “equal ending”. In Sweden where phone numbers with the country code consists of “12” digits (+46712345678) and without country code are of “10” digits (0712345678).

We can see that last “9” digits of both phone numbers are exactly same, so we define heuristic value for such kind of correspondence as:

We used Normalized edit distance for finding the similarity matrix, so minimum number of operations required to transform a “10” digit string into a “12” digit string provided that last “9” digits are similar is 3 and 12 is the length of the long string which are compared. By using term similarity formula as defined above in the term similarity section we got result 0.75. There is another case if a number is in the form like “0046712345678”, the length of the number is “13” digits and term similarity will be “0.76”.

From the above calculations we have seen for “equal ending” in both cases the value is “0.75” and “0.76” as we have millions of entries and due to error final similarity score may vary little bit, so we define an interval between 0.65 to 0.78. All the values in similarity matrix that are between 0.65 to 0.75 will test for correspondence type “equal ending”.

4.2.4.4 Contain In

In our practical observation we have seen in most of the cases the first or last or may be both name lies in the email of the person. As similarity score depends on the length of the string, to define interval for correspondence type “contain in” we considered different examples with different names and domain lengths (Gmail, hotmail, yahoomail.com, etc).

If a person has “3” character name (Ali) then with the domain its length will be “19” character (ali.ahmed@gmail.com) by using term similarity the final similarity score between name and email address is “0.17” and if the length of a name is “8” (Anderson) with domain its length become “26” characters (james.anderson@hotmail.com) then final similarity score is “0.30” and if we suppose that a person’s name length is “20” character then with the domain its length will be “49” character then the final similarity score will be “0.41”.
From the above calculation we have seen correspondence type contain in lies between intervals 0.17 to 0.41. We defined an interval between 0.15 to 0.45 for “contain in”. All the attributes values in similarity matrix that lies within this interval will test for correspondence type “contain in”. We will use a fast string matching Boyer-Moore algorithm to find either first attribute contain in second or second attribute contain in first [43].
CHAPTER 5

5 EMPIRICAL EVALUATION

5.1 Experiment

We present our experiment design followed by the execution of the experiment in the chapter. At the end of the chapter, result from the experiment is presented.

According to William R. Shadish an experiment is “A test under controlled conditions that is made to demonstrate a known truth, examines the validity of a hypothesis, or determines the efficiency of something previously untried”[41].

Proposed algorithm is having two main parts first is finding row similarity and second is finding attributes correspondence. In row similarity part algorithm finds the records that represent the same real world entity in two different heterogeneous data sets and the second part finding the correspondence between the attributes of records that are identified in the row similarity part. Attributes correspondence will be correct and effective only when it is calculated using the records that represent the same real world entity identified during the row similarity part. To measure the correctness and effectiveness of the proposed algorithm we evaluated it on three criteria.

1. How effective is the proposed algorithm in finding the pair of rows that represent the same real world entity when we are deciding row similarity based on the exactly one overlapping attributes which we called as potentially primary key in both data sets under the condition where schema design information is unknown?
2. How effective is the proposed algorithm in finding types of the attributes correspondence of the given pairs of rows which represent the same real world entities identified in row similarity part?
3. How effective is the proposed algorithm in finding types of the attributes correspondence without evaluating whole datasets?

5.1.1 Experimental Environment

To answer these questions we performed experiment using known data at the Ericsson controlled environment. All the experiments were performed by using HP Elite Book 8540w with an Intel core i5 processor, 4 GB Ram and window 7. The implementation of the proposed algorithm is done by using java 6 standard editions.

5.1.2 Experimental Test Data Generator

We used Ericsson’s data generator tool to create more realistic data. We can modify tool according to the requirement to generate desired data and to inject the rows with intentionally overlaps attributes. For each experiment we generated two CSV files CSV_FILE_1 as first dataset and CSV_FILE_2 as second dataset, each with 10000 rows.
5.1.3 Result Evaluation Measures

To evaluate results we used two well know and mostly used measures from information retrieval field precision and recall [42]. To measure the quality of the types of correspondence found by the proposed algorithm first need to manually find the type of correspondence between the attributes of heterogeneous data sets. The manually obtained real results can be used as standard to assess the quality of the results automatically determined by the proposed algorithm. Comparing the algorithm’s derived results with the real manually calculated results in the sets as shown in the figure below that can be used to define quality measures for attributes correspondence.

![Diagram showing actual vs retrieved same rows]

**Figure 5.1:** Manually and automatic calculated correspondence

There are four different condition for calculating the precision and recall.

- **False negatives (A):** Candidate pairs that may not be declared to be same while in fact they are.
- **True positives (B):** Candidate pairs that are correctly declared to be same.
- **False positives (C):** Candidate pairs that are declared to be same but in fact may not be same.
- **True negatives (D):** Candidate pairs that are correctly recognized as not being common.

Both false positive and false negative reduce the match quality. Based on the cardinality of these sets precision and recall can be computed.

\[
\begin{align*}
\text{Precision} & = \frac{\text{True positives}}{\text{True positives} + \text{False positives}} \\
\text{Precision} & = \frac{|B|}{|B|+|C|} \\
\text{Recall} & = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}} \\
\text{Recall} & = \frac{|B|}{|B|+|A|}
\end{align*}
\]

When precision and recall are equal to one it is an ideal case where no false negative and no false positive exist. Neither precision nor recall can be use alone to assess the derived attributes correspondence quality. Precision and recall sometimes miss leads, false positives are avoided leading to high precision. On the down side, this approach minimizes recall because of the high number of false negatives. To overcome this problem we use a tradeoff between precision and recall, the F-measure is often used in information retrieval field. It is the harmonic mean of precision and recall.
\begin{itemize}
  \item \( F - \text{measure} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \)
  \item \( F - \text{measure} = \frac{2 \times |B|}{(|A|+|B|)+(|B|+|C|)} \)
\end{itemize}

To achieve good F-measure values, both precision and recall must be high. With the typical tradeoff between precision and recall, both must have the same value to maximize the harmonic mean. Where F-measure close to 1 is its best value and close to 0 is its worst value.

To make our results more accurate each experiment performed three times with newly created data and by injecting similar rows at different positions in both datasets. The average results are reported at the end.

5.2 Row Similarity

How effective is the proposed algorithm in finding the pair of rows that represent the same real world entity when we are deciding row similarity based on the exactly one overlapping attributes which we called as potentially primary key in both data sets under the condition where schema design information is unknown?

To answer this question we performed Experiment 1(5.2.1) and Experiment 2(5.2.2).

5.2.1 Experiment 1

In the first experiment we want to know that our algorithm able to find the rows that are having similarity based on exactly one attribute common in both data sets which we called as potential primary key in both datasets. By using Ericsson test data generator we generated first dataset consists of twelve attributes and second dataset consists of five attributes as shown in the table below.

<table>
<thead>
<tr>
<th>Attributes of 1st Dataset</th>
<th>Attributes of 2nd Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security No.</td>
<td>Personal No.</td>
</tr>
<tr>
<td>First Name</td>
<td>---</td>
</tr>
<tr>
<td>Last Name</td>
<td>---</td>
</tr>
<tr>
<td>Street Address</td>
<td>---</td>
</tr>
<tr>
<td>Area Code</td>
<td>---</td>
</tr>
<tr>
<td>City Name</td>
<td>---</td>
</tr>
<tr>
<td>Language</td>
<td>Language</td>
</tr>
<tr>
<td>Phone No.</td>
<td>---</td>
</tr>
<tr>
<td>Cell No.</td>
<td>Cell No.</td>
</tr>
<tr>
<td>Profession</td>
<td>---</td>
</tr>
<tr>
<td>Gender</td>
<td>Gender</td>
</tr>
<tr>
<td></td>
<td>Email</td>
</tr>
</tbody>
</table>

Table 5.1: Attributes of DS 1 & DS 2

Phone number and cell number values are randomly generated and all other attributes values were randomly selected from the predefine list of values. For making rows similarity challenging we intentionally included two overlap attributes gender and language in both datasets and their values are selected from the same domain. Cell No. in dataset one is with country code and in second dataset without country code so they are different.
5.2.1.1 Row Similarity Detection

To show how effective and concrete is the row similarity “TFIDF($r_1, r_2$)” measure in detecting rows that are having similarity due to common key in both datasets without schema information, we performed precision and recall analysis. We injected 50,100,150 pair of rows representing same real world entities having only one attributes common, which is Social Security No. in first dataset and Personal No. in the second dataset for experiment one, two and three respectively. The average result shown in the figure 5.2 below by independently executing three experiments with newly created datasets for each experiment. The figure based on the calculation as mentioned in appendix (9.1).

![Figure 5.2: Row Similarity Result](image)

We achieved 100% precision in all three experiments which shows that row similarity measure of proposed algorithm efficiently distinguished rows that having similarity based on the common key from those rows that are not. Two attributes language and gender are common in both datasets having values from the same domain, there are so many rows that are having similarity because of these two attributes but not selected due to IDF, as these attribute are frequently appeared in the sample so their normalized weight are comparatively low than the other attributes and for row similarity measure we are considering only those attributes that are having maximum normalized weight, this is the reason that such pair of rows are not selected.

5.2.2 Experiment 2

As shown in the results of first experiment algorithm correctly identified the rows, having similarity based on the potential primary key from the rows that are not. Even two attributes language and gender are common in both datasets. Our algorithm considers similarity between the pair of row only when the common attributes of these rows having maximum normalized weights. In some cases it is possible that our algorithm selects a pair of row as similar even when similarity measure is not based on the potential primary key. For example we have two rows R and S, R from a sample of first dataset and S from the sample of second dataset.
Suppose we have sample size 50 rows, the attributes male and French appeared in the sample frequently and they having low IDF as a result the normalized weights of these attributes will be low so our algorithm will not consider these attributes in row similarity measure but attribute with value “atif” appeared only once in the sample of both data sets, so the normalized weight of these attributes will be equal to the weight of the primary key that are 1010 and 1001 in both rows. So our algorithm will also select such pair of rows as similar. To check the affect of these false positive on precision and recall we performed this experiment.

5.2.2.1 Row Similarity Detection

By using Ericsson test data generator we generated first data set consists of 15 attributes and second dataset consists of 8 attributes as shown in the table below.

<table>
<thead>
<tr>
<th>Attributes of 1st Dataset</th>
<th>Attributes of 2nd Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security No.</td>
<td>Personal No.</td>
</tr>
<tr>
<td>First Name</td>
<td>First Name</td>
</tr>
<tr>
<td>Last Name</td>
<td>---</td>
</tr>
<tr>
<td>Street Address</td>
<td>---</td>
</tr>
<tr>
<td>Area Code</td>
<td>---</td>
</tr>
<tr>
<td>City Name</td>
<td>---</td>
</tr>
<tr>
<td>Language</td>
<td>Language</td>
</tr>
<tr>
<td>Phone No.</td>
<td>---</td>
</tr>
<tr>
<td>Cell No.</td>
<td>Cell No.</td>
</tr>
<tr>
<td>Profession</td>
<td>---</td>
</tr>
<tr>
<td>Gender</td>
<td>Gender</td>
</tr>
<tr>
<td>Services</td>
<td>Email</td>
</tr>
<tr>
<td>Internet billing</td>
<td>Package</td>
</tr>
<tr>
<td></td>
<td>Prepaid</td>
</tr>
</tbody>
</table>

Table 5.2: Attributes of DS 1 && DS 2

The attribute name is also present in both datasets along with language and gender. We are using predefined list of values for language, gender and name and their values are selected from the same domain. Language and gender consist of few values so these values will repeat again and again in the datasets causing their IDF high consequently normalized weight of these attributes will be low and will not consider in the row measure. But the data domain for name attribute consists of long list of names there is possibility that some names will not repeat in the sample, the normalized weight of such attributes will be maximum or equal to the other attributes in same row. The rows with such attributes will also be selected in row similarity. To test it, we injected 100 pair of rows representing same real world entities having one attributes common, which is Social Security No. in first dataset and Personal No. in the second dataset. The average result shown in the figure 5.3 below by
independently executing three experiments with newly created datasets for each experiment. The figure based on the calculation as mentioned in appendix (9.2).

![Figure 5.3: Row Similarity Result with false positive](image)

The result presented in the figure shows that precision is decreasing as false positive are increasing. Decrease in precision will also affect the attribute correspondence because false positive representing different matching than the rows that are selected based on the common key. To overcome this problem we can use previous knowledge about the attributes correspondence to improve row similarity measure as suggested by Alexander Bilke and Felix Naumann [31]. One thing that is known about the attribute correspondence that both datasets having one field common we called it potential primary key in both datasets. We can use potential key in row similarity measure to correctly identify the pair of rows that are representing same real world entity. All the pair of rows that are selected as similar we will keep record of common attributes of the selected pair of rows. After finding all similar pair of rows we will consider only those pair of rows for term similarity which are selected on the bases of common attributes that are appeared in most pair of rows.

Now to remove the false positives we consider two datasets consist of sets of rows R and S respectively. Where \( R = \{ r_1, r_2, r_3, \ldots, r_m \} \) and \( S = \{ s_1, s_2, s_3, \ldots, s_n \} \). Each row consist of number of terms \( r_i = \{ a_1, a_2, a_3, \ldots, a_m \} \) and \( s_j = \{ b_1, b_2, b_3, \ldots, b_n \} \). Let A be a set having all the pair of rows selected as similar including false positive along with a set B having information about the position of the common attributes of selected pair of rows. We define A and B in the following way

\[
A = \{ (x_i, y_j) \mid x_i \in r_i \& y_j \in s_j ; i = 1,2,\ldots,m \& j = 1,2,\ldots,n \}
\]

\[
B = \{ (x_i, y_j) \mid x_i \in a_i \& y_j \in b_j ; i = 1,2,\ldots,m \& j = 1,2,\ldots,n \}
\]

Then we define a set C which gives us an order pair of positions of the common attributes which appears maximum times in B which is represented by \( \lambda \) and is given by

\[
C = \{ (x_i, y_j) = \lambda \mid \lambda \in B \}
\]

where B is the information about the position of the common attributes
Finally we define a set $D$ which actually gives us exact matching pair of rows after reducing the false positives and is given by

$$D = \{(x_i, y_j) \mid \forall B \approx \lambda \Rightarrow (x_i, y_j) \in A; x_i \in r_i \& y_j \in s_j; i = 1, 2, \ldots m \& j = 1, 2, \ldots n\}$$

By implemented above formula we removed false positive from the set of similar pair of rows. Only those pair of rows remains in the set which represent the same real world entity and their similarity based on the potential primary keys. Consequently a set of similar pair of rows without false positive will increase overall precision and f measure as shown in the figure 5.3.1 based on the calculation as mentioned in appendix (9.3).

![Figure 5.3.1: Row Similarity Result without false positive](image)

5.3 Term or Field and Attribute Similarity

How effective is the proposed algorithm in finding types of the attributes correspondence of the given pairs of rows which represent the same real world entities identified in row similarity part?

To answer this question we perform experiment 3 (5.3.1).

5.3.1 Experiment 3

In this experiment we will test how effective is our approach “Attribute Similarity” of finding the types of correspondence between the attributes. Effectiveness and correctness of the attributes correspondence depend upon row similarity phase. If row similarity phase correctly avoid false positive from the similar rows (true positive) based on the common key, then resultant attributes correspondence will be correct and effective. As we have discussed effectiveness and correctness of row similarity in previous section. By using Ericsson test data generator we generated first dataset consists of twelve attributes and second dataset consists of five attributes as shown in the table 5.1. The attribute cell number in first dataset having value included with country code and attribute cell number in second dataset without country code. Gender attribute present in both datasets extracting value from same domain. Attribute email in second dataset contains first and last name from first dataset.
Term similarity measure gives us the correspondence between the given attributes in the form of double number between 0 and 1. To find which types of correspondence exists between these attributes, correspondence values will compare with defined heuristic. We executed experiment by injecting 30 pair of rows in both dataset having same social security number and personal number at different position along with 6 false positive. The result is presented in the table below.

<table>
<thead>
<tr>
<th>DATASET 1</th>
<th>DATASET 2</th>
<th>Cell No.</th>
<th>Personal No.</th>
<th>Language</th>
<th>Email</th>
<th>Gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Name</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“B_CONTAIN_A”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>Last Name</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Define”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>Street Address</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>Area Code</td>
<td>“Not Define”</td>
<td>“Not Define”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>City Name</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Define”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>Language</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>Phone No.</td>
<td>“Not Define”</td>
<td>“Not Define”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>Profession</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Define”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>Services</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Define”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
</tr>
<tr>
<td>Gender</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Not Equal”</td>
<td>“Equal”</td>
<td>“Equal”</td>
</tr>
</tbody>
</table>

Table 5.3: Term Similarity Result

The correspondence types between the attributes found by the proposed algorithm are compared manually with the datasets that was used as input for the algorithm. We found proposed algorithm correctly identified types of correspondence between the attributes of the datasets.

5.4 Experiment 4

How effective is the proposed algorithm in finding types of the attributes correspondence without evaluating whole datasets?

To answer this question we performed experiment 4 (Section 5.4).

It is a time consuming process to completely evaluate the both datasets in search of similar rows. Instead of finding all the similar rows only few are needed to find the correspondence between the attributes [31, 48, 51]. To find how many similar rows are needed to find the types of correspondence, we performed five experiments. In the first experiment we injected 50 similar pairs of rows in both datasets along with the false positive at random positions. We searched for 40 similar pairs of rows to find the correspondence between the attributes. Similarly in second, third, fourth and fifth experiments, we injected 100, 200, 250,300 and searched for 60, 130, 180, 250 similar pairs of rows to find the correspondence. The results of these experiments are shown in the figure 5.4 below based on the calculated precision, recall and f-measure listed in Appendix (9.4). In the figure N is representing the number of injected similar rows and K is representing the number of similar rows used to find the correspondence between the attributes. On average, we found 90.4 percent f-measure of all experiments. From experiments we observed that choosing a low number of similar pairs of rows are better than high because the number of false positive are also increasing while setting the number of pairs of rows as high to find the correspondence attributes. Selecting a small number of pairs of rows to find corresponding attributes is not disadvantageous because our algorithm can find more if needed. Its takes less time to find few similar pairs of rows adds another advantage.
Figure 5.4: Precision, Recall and F-Measure
CHAPTER 6

6 DISCUSSION

In this research study, we introduced an instance based approach for finding the types of correspondence between the attributes of heterogeneous datasets. Since we are given large data sets, it was practically impossible to bring all data from both data sets into the main memory and perform operations to find the types of correspondence between the attributes of both data sets. To deal with this problem we used two sliding windows of flexible size, one for the first data sets and other for second data set. Only the data that were inside the sliding windows was compared in search of the similar pairs of rows. Windows size are flexible in sense, initially both windows extract 100 rows from each data sets and in case when no record found similar with in size 100, sliding windows will add 100 more rows from each datasets in this case windows size will become 200 rows.

Our approach consists of multiple steps: Row Similarity, Term or Field Similarity, Attribute Similarity and Types of correspondence. Row similarity is the process of evaluating the degree of similarity between the pairs of rows across different data sets to determine whether they refer to the same real world entity or not. The algorithms that are used for duplicate detection consider a pair of row as duplicate if maximum numbers of their attributes are overlapped [31, 34, 49]. In our problem both datasets share a potential primary key with the same format but it is unknown which columns are primary keys. Deciding a pair of row as duplicate based on the one field (potential primary key) common was very challenging under the circumstances when schema design information is unknown it is unclear which field in one row to compare with which field in the other and many fields values like gender (M/F) are coincidently common in many heterogeneous databases which make row similarity even harder.

In order to test effectiveness and correctness of the algorithm for Row Similarity, an experiment was designed and performed. We found that F-measure is decreasing when false positive are increasing. Decrease in f-measure will also affect the attribute correspondence because false positive representing different matching than the rows that are selected based on the common key. To overcome this problem we used known information about the datasets that they are sharing a common key. By using known information we proposed an approach to eliminate false positive from the candidate set. After have eliminated false positives an experiment was performed and showing improved results.

In the term similarity phase we calculated the term similarity matrices for each pair of rows. We calculated attributes similarity by taking average of each term similarity matrices. To test attributes similarity and types of correspondence we performed experiment number 3 with dataset 1 having twelve attributes and dataset 2 having five attributes. We found that our algorithm correctly identified the types of correspondence between the attributes of the datasets.

Since we have large datasets, finding the types of correspondence is a time consuming process. In experiment 4 we observed that only few pairs of rows are needed to find the types of correspondence between the attributes of heterogeneous data sets, reducing the required processor time significantly. Selecting a small number of pairs of rows for finding types of correspondence is not disadvantageous because our algorithm can find more if needed.
Although the calculated results are on a satisfactory level, there are things that need to be improved. Sliding windows change parameters even on the detection of false positives, which are also considered as similar pairs of rows in beginning. Hence some similar pairs of rows may be missed, even though by introducing a method based on the known correspondence efficiently removes the false positive. But still one needs a method to make sure that the pairs of rows selected to find the corresponding attributes are based on the common keys even when known correspondence information is not available.

In experiment 4, the variation in F-measure as shown in the figure 5.4 is due to false positives selected in the row similarity phase. In the row similarity phase a pair of row selected as similar based on two conditions. First, if pair of row has at least one overlapping attributes. Second overlapping attributes have equal or greater normalized weight among all the other attributes in their respective rows. In some cases, those pairs of rows are also selected as similar where there similarities are not based on the common primary keys as described in section 4.2.1 called as false positive. The F-measure values depend on the positions of the false positive in data sets. Because of this dependency F-measure values varies with different data sets as shown in the figure 5.4. The types of correspondence between the attributes not affected with the variation in F-measure values because after removing false positive by using proposed method as described in section 4.2.1, only those pairs of rows forwarded for term similarity phase that are similar based on common keys.

The proposed approach is designed to supplement existing research by enabling entity and attribute identification in situations where schema design information is unknown and both data sets share potential primary keys. The results from empirical analysis suggest that the method performs correctly for its intended purpose, i.e identifying the same entities and types of attributes correspondence between the heterogeneous data sets. But, the approach also has some limitations. Firstly, it cannot be employed on data sets that are not sharing the common potential primary keys and if both data sets are not in sorted order, i.e. the rows occur in lexicographic order with respect to the common key. Secondly, Attribute Similarity phase follows Row Similarity phase and no human interaction required. In some situations too many false positive might be detected, resulting in poor algorithm performance and it will also affect the attribute similarity.

6.1 Validity Threats

During any research validity assessment is an important factor and should be considered at the beginning of the research [51]. In [51], the authors identified four different types of validity threats i.e. internal validity, external validity, construct validity and conclusion validity.

- **Internal Validity**

Internal validity is used to deal with the biasness during the design and execution of the study [7, 51]. In the beginning qualitative research was started by conducting literature review. During literature reviews, the databases mentioned in the section 3.1.2 are used. Only those articles selected which were related to the approaches that are used to find the corresponding attributes from the heterogeneous databases. While conducting experiments the datasets that are used as inputs produced by using Ericsson data generator tools to make it more realistic. To eliminate the element of biasness and to create more realistic data our external supervisor generated the test data which were more realistic and according to the needs of his organization.
- **External Validity**

The external validity is concerned with generalization of study and internal validity is acts as prerequisite for the external validity [7, 51]. The treats on external validity are the conditions that limit the possibility of generalizing the study result that is deviated from the realistic environment [51]. To mitigate external validity threats each experiment performed three times with newly created data and injected similar rows at different positions of both datasets. The average results are presented to make it closer to the realistic results.

- **Construct Validity**

Construct validity is concerned with relationship between the design and the results that are concluded at the end of the research study [51]. To mitigate this threat, before conducting research study proper planning was done as mentioned in section 1.5.

- **Conclusion Validity**

Conclusion validity is deals with the conformability of the research study [51]. To mitigate this threat, the results obtain from literature review and experiments are critically reviewed before coming to conclusion. The experimental results were verified via various meeting with internal and external supervisors and the technical persons who were working at Ericsson environment.
CHAPTER 7

7 CONCLUSIONS

In this thesis we have introduced an instance based approach to find types of correspondence between the attributes of heterogeneous datasets. The approach, that is based on the simple idea of using similar pairs of rows to find the types of correspondence in datasets, identified the major obstacles in achieving the good results. One major obstacle is to find the similar pairs of rows among datasets with unmatched and only primary key overlapping attributes. We have presented a Row Similarity matrix based on the token based approach called cosine similarity that successfully identifies the similar pairs of rows from heterogeneous datasets where their similarity based on the common primary keys. Next we presented an Attribute Similarity matrix based on character based approach called levenstine edit distance that successfully identifies the corresponding attributes. At the end we found the types of correspondence between the corresponding attributes by using predefined heuristics values for each types. The analysis showed that the proposed approach was feasible to be used and it provided users a mean to find the corresponding attributes and the types of correspondence between them, based on the information extracted from the similar pairs of rows from the heterogeneous data sets where their similarity based on the same primary keys values.

7.1 Answers to the Research Questions

What is the current state of art for identifying attribute correspondences in heterogeneous databases?

The authors identified a combination of six state of the art of finding attributes correspondence between the heterogeneous databases by literature review as listed in section 3.2. These approaches are i) By Using Attributes Names ii) By Comparing Attributes Values iii) By Comparing Attributes Pattern iv) By Comparing Attributes Field Specification iv) Character Based Field Matching v) Token Based Field Matching vi) Q Gram Based Field Matching.

How do we find types of correspondence between the attributes of different data sets when schema design information of the data sets is unknown?

The authors proposed an instance based approach to find the correspondence between the attributes of heterogeneous datasets when schema design information is unknown as described in section 4.0. The proposed approach is validated by conducting an experiment in a controlled environment at Ericsson Karlskrona.

7.2 Future Work

The efficiency of the proposed approach can be improved when the both databases share the same primary but in different format. Two potential improvements should be investigated. Firstly, we have not considered one to many relation and mapping between the attributes. Secondly, we have considered only two data sets to find the types of correspondence between the attributes. This can also be performed for multiple data sets.
8 REFERENCES


CHAPTER 8

9 APPENDIX

9.1 With no false positives

a) \( \text{Total} = 50, T.P = 50, F.P = 0, F.N = 0 \) then

\[
\text{Precision} = \frac{50}{50 + 0} = 1
\]

And

\[
\text{Recall} = \frac{50}{50 + 0} = 1
\]

Now we check our results with the help of F-measure as

\[
F - measure = \frac{2 \times 1 \times 1}{1 + 1} = 1 = 100\%
\]

b) \( \text{Total} = 100, T.P = 100, F.P = 0, F.N = 0 \) then

\[
\text{Precision} = \frac{100}{100 + 0} = 1
\]

And

\[
\text{Recall} = \frac{100}{100 + 0} = 1
\]

Now we check our results with the help of F-measure as

\[
F - measure = \frac{2 \times 1 \times 1}{1 + 1} = 1 = 100\%
\]

c) \( \text{Total} = 150, T.P = 150, F.P = 0, F.N = 0 \) then

\[
\text{Precision} = \frac{150}{150 + 0} = 1
\]

And

\[
\text{Recall} = \frac{150}{150 + 0} = 1
\]

Now we check our results with the help of F-measure as

\[
F - measure = \frac{2 \times 1 \times 1}{1 + 1} = 1 = 100\%
\]

9.2 With false positive

a) With 10 false positive

i. \( \text{Total} = 100, T.P = 88, F.P = 10, F.N = 2 \) then

\[
\text{Precision} = \frac{88}{88 + 10} = 0.90
\]

And

\[
\text{Recall} = \frac{88}{88 + 2} = 0.97
\]

Now we check our results with the help of F-measure as
\[ F - \text{measure} = \frac{2 \times 0.90 \times 0.97}{0.90 + 0.97} = 0.93 = 93\% \]

ii. \( \text{Total} = 100, \ T.P = 87, \ F.P = 10, \ F.N = 3 \) then

\[ \text{Precision} = \frac{87}{87 + 10} = 0.90 \]

And

\[ \text{Recall} = \frac{87}{87 + 3} = 0.96 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{2 \times 0.90 \times 0.96}{0.90 + 0.96} = 0.93 = 93\% \]

iii. \( \text{Total} = 100, \ T.P = 89, \ F.P = 10, \ F.N = 1 \) then

\[ \text{Precision} = \frac{89}{89 + 10} = 0.90 \]

And

\[ \text{Recall} = \frac{89}{89 + 1} = 0.98 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{2 \times 0.90 \times 0.98}{0.90 + 0.98} = 0.94 = 94\% \]

iv. \( \text{Average} \)

\[ \text{Precision} = \frac{90 + 90 + 90}{3} = 0.90 \]

And

\[ \text{Recall} = \frac{97 + 96 + 98}{3} = 0.97 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{93 + 93 + 94}{3} = 0.93 = 93\% \]

b) \( \text{With 20 false positive} \)

i. \( \text{Total} = 100, \ T.P = 72, \ F.P = 20, \ F.N = 8 \) then

\[ \text{Precision} = \frac{72}{72 + 20} = 0.78 \]

And

\[ \text{Recall} = \frac{72}{72 + 8} = 0.90 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{2 \times 0.78 \times 0.90}{0.78 + 0.90} = 0.84 = 84\% \]

ii. \( \text{Total} = 100, \ T.P = 71, \ F.P = 20, \ F.N = 9 \) then

\[ \text{Precision} = \frac{71}{71 + 20} = 0.78 \]

And
Recall = \frac{71}{71 + 9} = 0.88

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{2 \times 0.78 \times 0.88}{0.78 + 0.88} = 0.83 = 83\% \]

iii. \textbf{Total} = 100, \textit{T.P} = 70, \textit{F.P} = 20, \textit{F.N} = 10 then

\[ \text{Precision} = \frac{70}{70 + 20} = 0.77 \]

And

\[ \text{Recall} = \frac{70}{70 + 10} = 0.87 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{2 \times 0.77 \times 0.87}{0.77 + 0.87} = 0.82 = 82\% \]

iv. \textbf{Average}

\[ \text{Precision} = \frac{78 + 78 + 77}{3} = 0.78 \]

And

\[ \text{Recall} = \frac{90 + 88 + 87}{3} = 0.88 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{84 + 83 + 82}{3} = 0.83 = 83\% \]

c) \textbf{With 30 false positive}

i. \textbf{Total} = 100, \textit{T.P} = 56, \textit{F.P} = 30, \textit{F.N} = 14 then

\[ \text{Precision} = \frac{56}{56 + 30} = 0.65 \]

And

\[ \text{Recall} = \frac{56}{56 + 14} = 0.80 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{2 \times 0.65 \times 0.80}{0.65 + 0.80} = 0.72 = 72\% \]

ii. \textbf{Total} = 100, \textit{T.P} = 55, \textit{F.P} = 30, \textit{F.N} = 15 then

\[ \text{Precision} = \frac{55}{55 + 30} = 0.64 \]

And

\[ \text{Recall} = \frac{55}{55 + 15} = 0.78 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{2 \times 0.64 \times 0.78}{0.64 + 0.78} = 0.70 = 70\% \]

iii. \textbf{Total} = 100, \textit{T.P} = 54, \textit{F.P} = 30, \textit{F.N} = 16 then
\[
\text{Precision} = \frac{54}{54 + 30} = 0.64
\]

And

\[
\text{Recall} = \frac{54}{54 + 16} = 0.77
\]

Now we check our results with the help of F-measure as

\[
F - \text{measure} = \frac{2 \times 0.64 \times 0.77}{0.64 + 0.77} = 0.70 = 70\%
\]

iv. Average

\[
\text{Precision} = \frac{65 + 64 + 64}{3} = 0.64
\]

And

\[
\text{Recall} = \frac{80 + 78 + 77}{3} = 0.78
\]

Now we check our results with the help of F-measure as

\[
F - \text{measure} = \frac{72 + 70 + 70}{3} = 0.71 = 71\%
\]

9.3 After removing false positives

a) \(Total = 50, T.P = 47, F.P = 1, F.N = 2\) then

\[
\text{Precision} = \frac{47}{47 + 1} = 0.98
\]

And

\[
\text{Recall} = \frac{47}{47 + 2} = 0.96
\]

Now we check our results with the help of F-measure as

\[
F - \text{measure} = \frac{2 \times 0.98 \times 0.96}{0.98 + 0.96} = 0.97 = 97\%
\]

b) \(Total = 100, T.P = 95, F.P = 2, F.N = 3\) then

\[
\text{Precision} = \frac{95}{95 + 2} = 0.98
\]

And

\[
\text{Recall} = \frac{95}{95 + 3} = 0.97
\]

Now we check our results with the help of F-measure as

\[
F - \text{measure} = \frac{2 \times 0.98 \times 0.97}{0.98 + 0.97} = 0.97 = 97\%
\]

c) \(Total = 150, T.P = 141, F.P = 5, F.N = 4\) then

\[
\text{Precision} = \frac{141}{141 + 5} = 0.96
\]

And

\[
\text{Recall} = \frac{141}{141 + 4} = 0.97
\]

Now we check our results with the help of F-measure as

\[
F - \text{measure} = \frac{2 \times 0.96 \times 0.97}{0.96 + 0.97} = 0.96 = 96\%
\]
9.4 Different number of similar rows used for finding attribute correspondence

i. When top 40 pairs are detected
\[ N = 50, K = 40, T. P = 37, F. P = 3, F. N = 2 \]
Where \( N \) the total number of is similar rows and \( K \) is the number of similar rows used to find the correspondence between the attributes of two heterogeneous data sets. We find precision and recall as

\[
\text{Precision} = \frac{37}{37 + 3} = 0.92
\]

And

\[
\text{Recall} = \frac{37}{37 + 2} = 0.95
\]

Now we check our results with the help of F-measure as

\[
\text{F - measure} = \frac{2 \times 0.92 \times 0.95}{0.92 + 0.95} = 0.93 = 93% 
\]

ii. When top 60 pairs are detected
\[ N = 100, K = 60, T. P = 50, F. P = 7, F. N = 5 \]
where \( N \) the total number of is similar rows and \( K \) is the number of similar rows used to find the correspondence between the attributes of two heterogeneous data sets. We find precision and recall as

\[
\text{Precision} = \frac{50}{50 + 7} = 0.88
\]

And

\[
\text{Recall} = \frac{50}{50 + 5} = 0.91
\]

Now we check our results with the help of F-measure as

\[
\text{F - measure} = \frac{2 \times 0.88 \times 0.91}{0.88 + 0.91} = 0.89 = 89% 
\]

iii. When top 130 pairs are detected
\[ N = 200, K = 130, T. P = 120, F. P = 8, F. N = 4 \]
where \( N \) the total number of is similar rows and \( K \) is the number of similar rows used to find the correspondence between the attributes of two heterogeneous data sets. We find precision and recall as

\[
\text{Precision} = \frac{110}{110 + 8} = 0.93
\]

And

\[
\text{Recall} = \frac{110}{110 + 4} = 0.96
\]

Now we check our results with the help of F-measure as

\[
\text{F - measure} = \frac{2 \times 0.93 \times 0.96}{0.93 + 0.96} = 0.94 = 94% 
\]

iv. When top 180 pairs are detected
\[ N = 250, K = 180, T. P = 140, F. P = 30, F. N = 15 \]
where \( N \) the total number of is similar rows and \( K \) is the number of similar rows used to find the correspondence between the attributes of two heterogeneous data sets. We find precision and recall as

49
\[ \text{Precision} = \frac{140}{140 + 30} = 0.82 \]

And

\[ \text{Recall} = \frac{140}{140 + 15} = 0.90 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{2 \times 0.82 \times 0.90}{0.82 + 0.90} = 0.86 = 86\% \]

v. When top 250 pairs are detected

\[ N = 300, K = 250, T.P = 231, F.P = 26, F.N = 19 \]

where \( N \) the total number of is similar rows and \( K \) is the number of similar rows used to find the correspondence between the attributes of two heterogeneous data sets. We find precision and recall as

\[ \text{Precision} = \frac{231}{231 + 26} = 0.89 \]

And

\[ \text{Recall} = \frac{231}{231 + 19} = 0.92 \]

Now we check our results with the help of F-measure as

\[ F - \text{measure} = \frac{2 \times 0.89 \times 0.92}{0.89 + 0.92} = 0.90 = 90\% \]