經營學博士學位論文

Toward Efficient and Accurate Schema Matching - Cross Similarity Vector Approach and Learning-based Matcher Combination

효율적인 스키마 매칭 방법 – 교차 유사벡터 접근과 기계학습 기반의 매쳐 조합 전략을 중심으로

2014年 8月

서울大學校 大學院

經營學科 經營學 專攻

崔 榮 碩

Toward Efficient and Accurate Schema Matching - Cross Similarity Vector Approach and Learning-based Matcher Combination

지도교수 : 박 진 수

이 논문을 경영학 박사 학위 논문으로 제출함
2014년 4월

서울대학교 대학원

경영학과 경영정보 전공

최 영 석

최영석의 박사학위논문을 인준함
2014년 6월

위 원 장: 노 상 규 (인)

부위원장: 박 진 수 (인)

위 원: 유 병 준 (인)

위 원: 오 정 석 (인)

위 원: 이 동 원 (인)

To my wife, Anna…

Abstract

Toward Efficient and Accurate Schema Matching - Cross Similarity Vector Approach and Learning-based Matcher Combination

Youngseok Choi

Department of Management Information Systems

College of Business Administration

Seoul National University

Schema matching is one of the main challenges in in many database application domains, such as data integration, E-business, data warehousing, and semantic query processing. Over the past 20 years, different schema matching methods have been proposed and shown to be successful in various situations. Schema matching still seems to involve ad-hoc solutions with only a few works that involve foundational principles of schema matching because most of schema matching situations are too generic. Though many advanced matching algorithms have emerged, the schema matching research remains a critical issue. Different algorithms have been implemented to resolve different types of schema heterogeneities, including the differences in design methodologies, the naming conventions, and the level of specificity of schemas, among others.

Since hundreds of schema matching algorithms have been proposed, a strategy for combining existing matchers becomes one of the most important issues on schema matching studies. Composite and hybrid matching approaches are the main methodology to cope with the various schema matching situations. Selecting and combining appropriate matching algorithms for a given matching situation is very critical for improving matching performance.

Schema matching research can be classified into two folds; finding new single schema matching algorithms and making a strategy for combining multiple matchers. Individual matcher usually reflect the properties of schema element such as name, structure, constraints, etc. In this dissertation, first, I propose a novel approach to find structural similarity using the concept of cross similarity vector. Proposed approach has its theoretical foundation from a context in database design. The approach covers the drawbacks of existing structural measures. By calculating the similarity between context structures using cross similarity vector, more advanced structural schema matching metric can be found. Second, very efficient way to combine existing matchers is introduced and evaluated using the sample schema data. Most of existing combinational approaches have focused on finding optimal linear combination of multiple measures, which is a part of heuristic optimization. These approaches tend to make arbitrary weight and threshold by iterative test. This task is very inefficient and the complexity of problem increases when the number of matchers is large or increases. To solve this kind of problem, I suggest the matcher combination strategy based on supervised learning classifier.

By transforming the schema matching task into learning-based classification problem, the number of parameter is dramatically decreased as the number of matcher increases. Comparing representative schema matching prototype, proposed approach is fully automated. Any types of human intervention such as abbreviation processing, user feedback, are not adopted at all. The performance of proposed approach is also better than existing fully automatic schema matching algorithms and nearly at the level of semi-automatic schema matching approach.

Keywords: schema matching, data matching, combinational matching, matcher combination, cross similarity vector, learning-based classification, data integration, machine learning.

Table of Contents

Table Index

Figure Index

PART1. INTRODUCTION

CHAPTER1. Schema Matching Research – Its Motivation

Schema matching involves matching among concepts which describe the meaning of data in various heterogeneous, distributed data (Gal 2006). Schema matching is recognized as one of the basic operations required in the process of data integration (Bernstein & Melnik 2004). Data integration and exchange can be ssen as the process of converting an instance of one schema, called the source schema, into an instance of a different schema, called the target schema, according to a given specification (Hernandez et al. 2008). This problem necessitates finding correspondence among elements of given schemas. Matching tasks are closely related to the problem of defining global views of heterogeneous data sources to support querying and cooperation activities (Castano and De Antonellis 2001).

A schema is a formal structure of an engineered artifact, such as SQL schema, XML schema, entity–relationship diagram, ontology description, interface definition, and form definition (Bernstein et al. 2011). Manually specifying schema matches is obviously a tedious, time-consuming, error-prone, and consequently, expensive process. This problem is becoming worse given the rapidly increasing number of Web data sources and e-businesses to integrate (Rahm & Bernstein 2001). Therefore, numerous researchers have tried to find more

effective and efficient means of matching schemas automatically (Gong et al. 2012).

Schema matching has been a very active research area, particularly during the last decade in which hundreds of techniques and prototypes for automatic matching have been developed (Rahm & Bernstein 2001). Although numerous advanced matching algorithms have emerged, the schema matching research remains a critical issue and definitive solutions for schema matching are not yet available (Atzeni 2007). Various algorithms have been implemented to resolve different types of schema heterogeneities, including the differences in design methodologies, the naming conventions, and the level of specificity of schemas, among others (Batini et al. 1986). The algorithms for the matchers are usually too generic irrespective of the schema matching scenario. This situation indicates that a single matcher cannot be optimized for all matching scenarios. A matching algorithm cannot be effective in all scenarios. In the case of a large scale schema matching problem, improving the matching efficiency is more challenging.

CHAPTER2. The Definition of Problem Space for Dissertation

Existing schema matching studies have examined in various structural aspect of schema matching. Schema matching task can be defined as finding correspondence between schema elements using a **specific metric**[1] or **combining existing matching metric**. In this dissertation, the aim of dissertation will be focused on

[1] Matching measure, matching metric, matcher, and matching algorithms have had same meaning in past literatures.

suggesting new matcher and strategy for combining existing matchers.

2.1. Designing the matching metric

Matching metric is a necessary factor for performing schema matching task. Usually, a single metric returns the value between 0 and 1, which means how two schema element similar. The metric can be classified into name similarity and structural similarity. A name similarity is the metric which uses the information about the name of element; i.e. the semantic of element, properties as a string, etc. A structural similarity finds the similarity based on the structural property of each schema element; i.e. the morphological structure around the element, hierarchical structure of the element, etc. Each similarity has their own strength so that they are usually combined to make better performance.

The similarity metric based on elements' name has been studied since many researchers in computational linguistics and computer science have had interested. Many matching metrics have been used for schema matching very usefully and the metrics is very critical role in improving the performance of existing schema matching architecture. Most of metrics are designed for general purpose such as calculating similarity between two concepts. In this regard, there needs to be a name similarity metric for schema matching task. This research is completed in another working papers (Choi et al. 2014), so this is not the scope of this dissertation.

In this dissertation, a novel approach to design a structural measure which reflect the semantic structure of element. By combining structural property and name property, the weaknesses of structural property can be minimized. This is the first contribution of this dissertation and details are presented in Part 3.

2.2. Making strategy for combination of various matcher

Since the schema matching research have been started, hundreds of schema matching algorithms have been proposed in both research and practical community. Each of existing schema matching has their own strengths for specific situation and single matching algorithm cannot perform well in every matching situation because schema matching task is too generic[2]. This is the reason most of representative approaches adopt the combination strategy for schema matching. Combining various matching algorithms can be seen as the complex optimization problem to find optimal combination; i.e. finding the weights of individual matcher for best matching performance (Gottlob et al. 2011).

In this dissertation, this combination strategy is simplified by adopting machine learning-based classification problem. I transform the schema matching task into learning-based classification to simplify the overall process of matching. The details of this idea and experimental evaluation will be presented in Part 4.

CHAPTER 3. Theoretical Foundations of Schema Matching Research

Several attempts at setting theoretical foundations for schema matching exist in literatures. The theoretical aspect of schema matching research can be divided into model management and operation matching. Most theoretical foundations are

[2] 'generic' indicates the variety of each schema matching tasks; every matching has their own property so that this situation cannot be covered by single matching algorithm.

related to symbolic algebra. The theoretical justification of matching space reduction also can be found in existing research (Smiljanić et al. 2006, Bellahsene et al. 2011). We describe the efficiency of matching space reduction using simple arithmetic proof in last subsection.

3.1. Model Management

Model management is a framework for supporting applications related to metadata, wherein models and mappings are manipulated as first-class objects using various operations (Bernstein & Rahm 2000). The original idea of model management may be closely connected the fundamental problem of modern database management such as semantic heterogeneity (Hull 1997). A model is a structure representing a designed artifact, such as an XML DTD and Web site schema, among others. A number of mathematical foundations make it easier to manage such models. Many examples of high-level algebraic operations are currently being used for specific metadata applications (Jannink et al. 2009; Miller et al. 1994; Mitra et al. 2000).

Bernstein et al. (2000) suggested key algebraic operations for model management. Fundamental operations include:

- Matching – taking two models as input and returning a mapping between them as output;
- Composing – taking two mappings as input and returning their composition as output;

- Merging – taking two models and a mapping between them as input and returning a model merging the two models (using the mapping to guide the merging) as output;
- Setting operations on models – involves unions, intersections, and differences; and
- Projecting and selecting models – which are comparable to relational algebra.

These algebraic operations manipulate models and their mappings, each of which connects the elements of two models. A matching operation, which is one of the most important operations in model management, can be defined in a more formal manner (The details of matching operations are discussed in the next subsection).

To show hot these operations may be used, Bernstein (2001) consider the problem of populating a data warehouse. Suppose that we have a model S_1 of a data source and a mapping map_{1W} from S_1 to a model W of a data warehouse. Now we are given a model S_2 of a second data source (see Figure below). We can merge S_2 into the data warehouse as follows: (1) Match S_2 and S_1, yielding a mapping map_{21}; (2) Compose map_{21} with map_{1W}, to produce a mapping map_{2W} from S2 to W. Step (1) characterizes those parts of S_2 that are the same as S_1.Step (2) reuses map_{1W} by applying it to those parts of S_2 that are the same as S_1. In Bernstein et al. 2000, they described a detailed version of this scenario and others like it, to show how to use the model management algebra to solve some practical data warehousing problems.

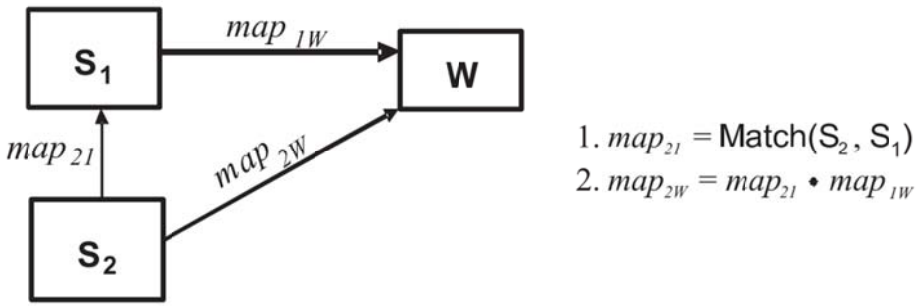


Figure 1. A Simple Data Warehouse Scenario Using Model Management (Bernstein 2001)

3.2. Matching Operations and Matching Process

The foundations of a matching operation in a schema matching process can be found in numerous related studies. Based on the research of Shvaiko and Euzenat (2005) and Euzenat (2004), a matching operation can be defined as a quintuple: <*id, e, e', n, R*>, where

- *id* is a unique identifier of a given mapping element;
- *e* and *e'* are the entities (tables, XML elements, properties, classes) of the first and second schema, respectively;
- *n* is a confidential measure in several mathematical structure holdings for the correspondence between entities *e* and *e'*; and
- R is a relation [equivalence (=), more general (⊇), disjointness (⊥), overlapping (∩)] holding between entities *e* and *e'*.

Based on this matching operation, we can define the matching process which determines the alignment (A) for a pair of schemas. Alignment is a set of mapping elements (Shvaiko and Euzenat 2005). The general matching process is depicted in Figure 1.

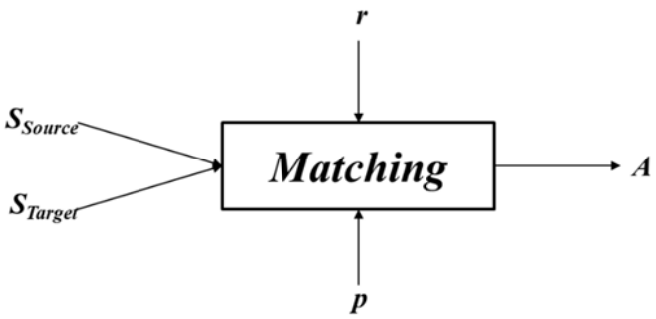


Figure 2. The matching process.[3]

The input of this matching process is a pair of schemas [source (S_{Source}) and target (S_{Target})]. External resources or auxiliary information (r) (such as thesauri, ontology) can be used by the matching process. To determine the result of the matching process (alignment, A), the relation among mapping elements is defined based on the matching parameters (p) in advance.

In this dissertation, I modified Shvaiko and Euzenat (2005)'s matching frame. To cope with combinational and hybrid matching scenario, matching parameter is

3 This figure was modified based on Figure 2 in Shvaiko and Euzenat 2005. The descriptions for this figure were also refined and modified.

atomized into weights for individual matcher and matching threshold. The matching model which will be utilized in this dissertation is depicted below.

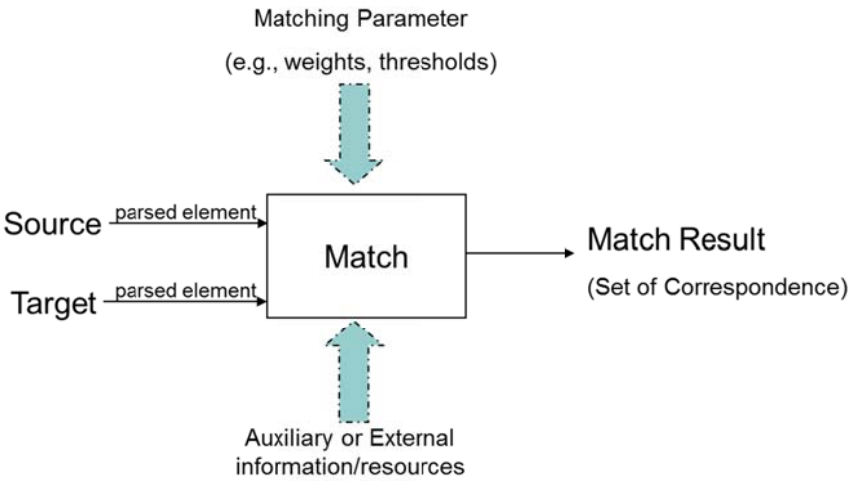


Figure 3. Process of schema matching process in this dissertation.

3.3. Matching Space Reduction

The standard approach for pairwise schema matching is to compare every element of the target schema with every element of the source schema to determine the matching schema elements. Such approach has at least a quadratic complexity with respect to the schema. Not only efficiency problems for large schema exist but also a large matching space makes identifying correctly the matching element pairs very difficult. Thus, large-scale schema matching tasks must reduce the matching space to improve the matching efficiency (Bellahsene et al. 2011).

For schema matching, the partition-based matching approach has been commonly considered to reduce the matching space. This approach increases the matching efficiency by performing partition-wise matching. The efficiency of the partition-based matching can be proven easily by arithmetic. In this part, the efficiency of the partition-based matching is proven.

Let the total number of element in the source schema be N_{Source}. If we assume that the source schema has mutually exhausted p partitions and the number of elements in each partition is n_{s1}–n_{sp} elements, N_{Source} can be expressed as

$$N_{source} = \sum_{i=1}^{p} n_{si}.$$

We let the total number of elements in source schema as N_{Target} and the number of its partitions as q. If the number of elements in each partition is n_{t1}– n_{tq}, then N_{Target} can be expressed in the same manner.

$$N_{Target} = \sum_{i=1}^{q} n_{ti}.$$

In the case of the standard approach, i.e., pairwise element matching, the total number of matching is

$$N_{Source} \times N_{Target} = \sum_{i=1}^{p} n_{si} \times \sum_{i=1}^{q} n_{ti}.$$

However, in the case of the partition-based matching, the total computation can be reduced to

$$\sum_{i=1}^{p} n_{si} \times \left(n_{tj} + n_{tk}\right), \text{ where } 1 \leq j,\ k \leq q \ [4].$$

Unless each partition of the source schema tries to match all the partitions of the target schema, the partition-based matching can have an advantage in the matching complexity.

[4] In this case, we assume that each partition has two matching candidate partitions of the target schema.

PART2. RELATED WORKS

CHAPTER4. The Classification of Schema Matching Research

The existing schema matching techniques are shown in some representative surveys of schema matching (Rahm and Bernstein 2001; Shvaiko and Euzenat 2005; Bernstein et al. 2011). They suggest a classification of individual matching techniques. An implementation of a schema-matching process may use multiple matching algorithms, thus allowing us to select individual matchers depending on the matching scenario. The following list shows the general classification of individual matchers and recent research belonging to each classification. The following categories are reorganized based on existing surveys[5] (Rahm and Bernstein 2001; Shvaiko and Euzenat 2005; Bernstein et al. 2011).

4.1. Rahm and Bernstein's Classification

Rahm and Bernstein (2001)'s classification of schema matching approach has given the research insight to related researchers. An implementation of Match operation may use multiple match algorithms or matchers. Matcher might be selected depending on the application domain and schema types. When it comes to use multiple matchers, there are two alternative to implement match operation. First, there is the realization of individual matchers, each of which computes a

[5] Related works before 1990s are well summarized in Batini et al. 1986.

mapping based on a single matching criterion. Second, there is the combination of individual matchers, either by using multiple matching criteria within an integrated hybrid matcher or by combining multiple match results produced by different match algorithms within a composite matcher (Rahm and Bernstein 2001). For individual matchers, Rahm and Bernstein consider the following largely-orthogonal classification criteria (see figure below).

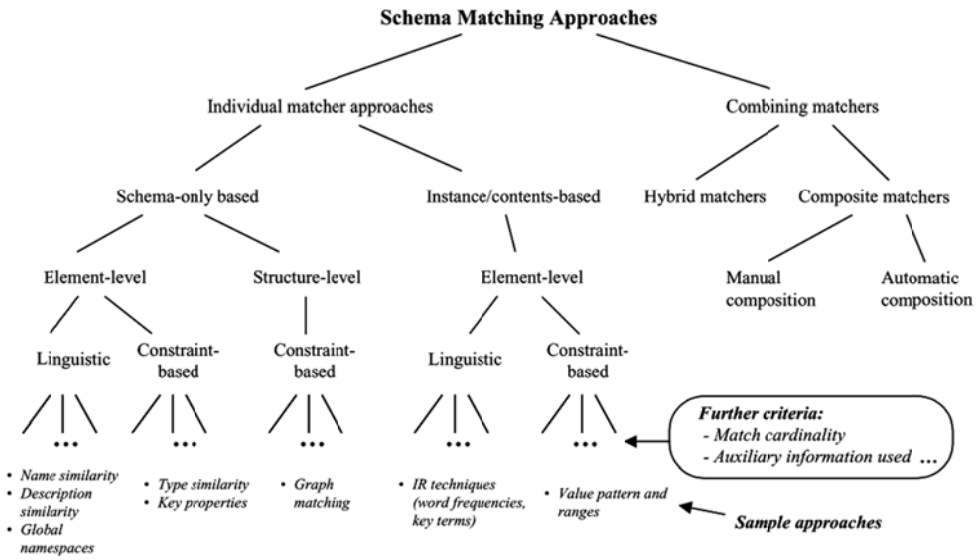


Figure 4. Classification of Schema Matching Approach (Rahm and Bernstein 2001)

4.2. Shvaiko and Euzenat's Classification

Another well-defined classification of schema matching approach can be found in Shvaiko and Euzenat (2005). They discuss only schema-based elementary matchers, that means only schema/ontology information is considered, not instance data1. To

ground and ensure a comprehensive coverage for their classification they have analyzed state of the art approaches used for schema-based matching. One of big difference from Rahm and Bernstein (2001)'s classification lies in their guidelines for building their classification

Exhaustivity. The extension of categories dividing a particular category must cover its extension (i.e., their aggregation should give the complete extension of the category);

Disjointness. In order to have a proper tree, the categories dividing one category should be pairwise disjoint by construction;

Homogeneity. In addition, the criterion used for further dividing one category should be of the same nature (i.e., should come from the same dimension). This usually helps guaranteeing disjointness;

Saturation. Classes of concrete matching techniques should be as specific and discriminative as possible in order to provide a fine grained distinction between possible alternatives. These classes have been identified following a saturation principle: they have been added/modified till the saturation was reached, namely taking into account new techniques did not require introducing new classes or modifying them.

Based on these four guidelines, they classify schema-based matching approach (see figure below).

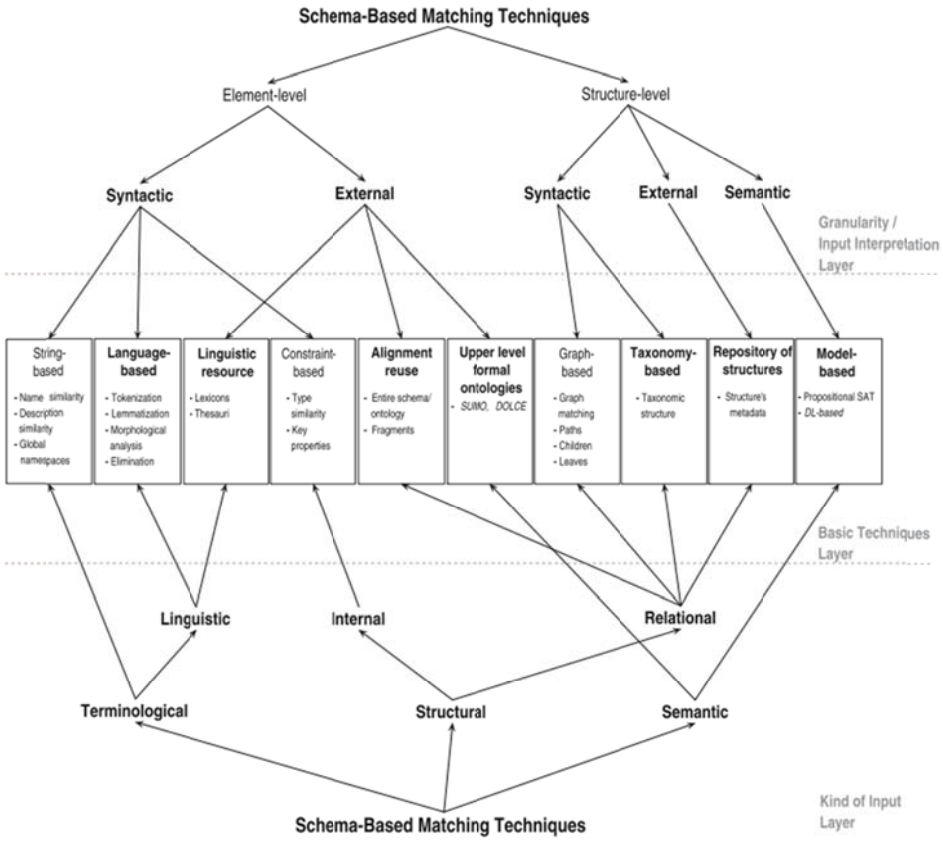


Figure 5. A retained classification of elementary schema-based matching approaches (Shvaiko and Euzenat 2005)

4.3. Details of Classification

In prior sections, we reviewed two representative survey of schema matching research. In this section, we will present the details of each classified approaches.

- **Instance vs. schema-based matching**: Matching approaches can consider instance data (Chua et al. 2003, Kang and Naughton 2003, Wang et al. 2004, Bilke and Naumann 2005, Kang et al. 2005, Zhao and Ram 2007, Jaiswal et al.

2010, Ding and Wang 2011, Jaiswal et al. 2013) or only the information of the schema element. Most of published research on schema matching belong to schema-based matching. One of major challenge of instance-based matching is how to evaluate proposed approach. To evaluate instance-based matching, real database schema with whole instance data should be given. However, to obtaining test schema with its instance is not common for usual test scenario. For this reason, most of research which belong to this category evaluate their approach using virtually generated data by users. Instance-based matching have adopted information theory-based measure such as entropy or mutual information. Computational linguistics-based measure also can be applied to instance data though it takes longer time in massive instance case.

Some of researcher argue that instance-based schema matching approach is more appropriate for real matching scenario because database schema of most firms has its corresponding instance in real world. In addition, instance data may be considered as additional data of schema element. If more information is given to matching, the matching problem is easier. To enable to realize many kind of instance-based approach, researchers should try to obtain real database schema and its instance.

- **Element vs. structure matching**: Matches can be performed for individual schema elements or for combinations of elements, such as complex schema structures. Structural matching mainly uses the topological approach (Melnik et al. 2002, Bertino et al. 2004, Lu et al. 2007, Algergawy et al. 2009, Fan et al. 2010, Gal 2012). The common

utilization of schema's structural aspect is to consider the parents, children, or the leaves of each schema element.

Do Hong Hai (2005) emphasized the importance of structural matching in his dissertation: "*In general, structural match approaches may be misled by structural conflicts, which occur due different detail levels of information represented by schema elements (p. 32)*". This statement does not mean single structural matcher guarantees the high matching performance. Few research adopt only structural matcher without other matcher such as element-based matcher. Structural matching algorithm has been frequently combined with other matching approach to implement multiple matcher approach, i.e. combinational and hybrid matching.

- **Language vs. constraint-based matching**: A matcher can use a linguistic-based approach such as those based on names and textual descriptions of schema elements (Giunchiglia et al. 2005, Madhavan et al. 2005, Casanova et al. 2007, Islam et al. 2008, Po and Sorrentino 2011, Chen et al. 2012) or a constraint-based approach such as those based on keys and relationships (Yu and Popa 2004, Smiljanić et al. 2005, Kim et al. 2007, Zhao et al. 2012).

Language-based (or linguistics-based approach) is one of most frequently used matcher in multiple matcher approach. This approach usually adopt two kind of matcher: semantic and non-semantic measure for string. In the case of non-semantic measure, the meaning of schema element is not considered for calculating similarity. Edit-distance, N-gram, and Soundex measure are representative non-semantic linguistic measure. Semantic measure usually

adopt semantic network and dictionary to calculate similarity between schema elements.

Schema often has constraints to declare data types, allowable values, value ranges, etc. This constraints can be used for finding correspondence between two elements. For example, if one is string and another is integer, they are less likely to correspond each other. This is the basic idea of constraints-based matcher.

- **Auxiliary information-based matching**: Most matchers do not only rely on input schemas S_1 and S_2, but also on auxiliary information, such as dictionaries, global schemas, previous matching decisions, and user input (Madhavan e al. 2005, Drumm et al. 2007, Islam et al. 2008, Cruz et al. 2009, Algergawy et al. 2010, Algergawy et al. 2011, Thang and Nam 2010). Corpus-based schema matching has its origin from corpus-based text similarity research (Mihalcea et al. 2006, Halevy and Madhavan 2003).

  Most popular semantic similarity measure use WordNet as their auxiliary information. WordNet is network-typed dictionary which provide the meaning of words and their synonym and hypernym. Previous studies on semantic similarity using semantic network can broadly be classified into: those based on the topological method and those based on the statistical method. The topological method uses various forms of semantic network or taxonomy to compute the semantic relatedness between concepts. The representative semantic networks based on the topological method are WordNet and Roget's thesaurus. On the other hand, the statistical method is a technique for

calculating the statistical association between concepts by using auxiliary data, such as a dictionary or corpus that is not in the form of a network.

Corpus-based schema matching approach have used corpus as a domain knowledge. Common type of corpus is set of news from specific domain related to matching scenario. Using this kind of domain corpus, user-defined semantic network can be implemented for calculating similarity between schema elements.

In addition to these categories, some schema matching based on algebraic approach methods can be found in past literature (Algebraic Matching).

- **Algebraic Matching:** Some studies have tried to solve the schema matching problem using pure and applied algebra. They transform the schema matching problem into algebraic problem and finding optimal solution for matching result (Bernstein & Rahm 2000, Sabetzadeh and Easterbrook 2005, Zhang et al. 2005, Zhang et al. 2008, Hosain and Jamil 2010).

  To implement algebraic matching, a formal meta-meta model to describe the various schema types should be defined. Many kind of schemas can be considered as meta-model. A meta-meta model is a representation language in which models and meta-models are represented (Zhang et al. 2008). Schema matching can be defined as algebraic manner. Zhang et al. (2008) define schema matching algebraically, "If S is the source schema, T is the target schema, the matching result of two schemas is a set $m \subseteq I_S \times I_T$ that contains every matched couple $<s, t> \in I_S \times I_T$ ".

- **Web 2.0 Approach:** McCann et al. (2008) proposes Web 2.0 approach that uses the knowledge of multiple users. Nguyen et al. (2010) and Nguyen et al. (2014) also introduce the web-based schema matching system to aggregate multi-users' matching.

We have reviewed several categories of individual matchers. To resolve the more general and complex schema matching problem, combining multiple matching approaches can be effective because the individual matchers may only be applicable and appropriate for a given specific match task. We call this approach "Combinational Matching" in this dissertation.

- **Combinational Matching[6]:** Castano et al. (1997)**,** COMA++ (Do et al. 2002; Aumueller et al. 2005), Cupid (Madhavan et al. 2001), and eTuner (Lee et al. 2007) are well-known schema matching systems, and these systems utilize multiple matching algorithms to deal with more complex matching tasks. Domshlak et al. (2007) proposes the matcher ensemble approach by aggregating ranking. Feng et al. (2010) suggests GATuner which tuning the matcher's parameter using generic algorithm. Jeong et al. 2008 implements XML schema matching algorithm using supervised learning algorithm. Generally, combinational matching shows better performance than the individual matcher.

Unfortunately, matching large schemas are very challenging based on reviewed studies because most previous studies have addressed small matching tasks only.

[6] Peukert et al. 2010 compares the combination method of multiple matchers.

Only a few studies tried to solve large-scale matching (He and Chang 2004; Rahm 2011).

CHAPTER5. Representative Schema Matching Architecture

5.1. COMA++

COMA++ is the representative schema matching architecture. COMA++ also provides match operation for ontology matching. It extends their previous prototype COMA (Do and Rahm 2002) utilizing a composite approach to combine different match algorithms. It comes with a GUI enabling a varous user interactions. Using a canonical data representation, COMA++ uniformly supports schemas and ontologies, e.g. W3C XML Schema and OWL. COMA++ includes ontology matching, in particular the utilization of shared taxonomies. Different match strategies can be applied including various forms of reusing previously determined match results and a so-called fragment-based match approach which decomposes a large match problem into smaller problems (Aumueller et al. 2005). Details of COMA++ architecture depicted in figure below.

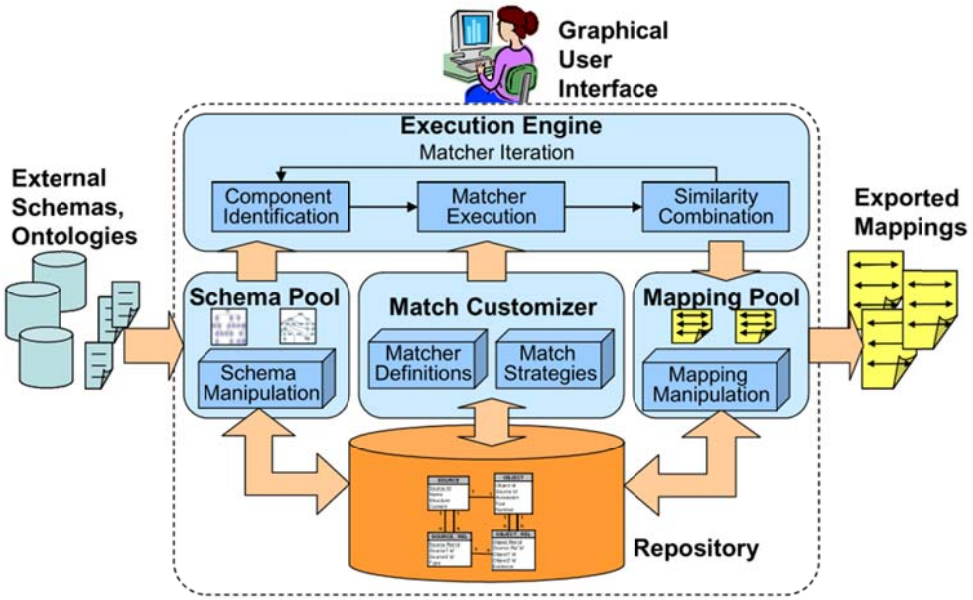


Figure 6. Architecture of COMA++ (Aumueller 2005)

COMA++ has four components: Model Pool, Execution Engine, Match Customizer, and Mapping Pool. The Model Pool provides different functions to import external schemas and ontologies, and to load and save them from/to the repository. COMA++ support most kind of schema-formed data types. Matching process is performed in the Execution Engine.[7] Matching Customizer supports the optimization process of iterative matching. Finally, the Mapping Pool maintains all generated mappings and offers various function for existing matching result. The

[7] In Aumueller et al. (2005), they insist COMA++ is automatic schema matching architecture. However, COMA++ is semi-automatic approach because of manual abbreviation process and interaction between user and architecture.

COMA++'s features are summarized in table below.

Table 1. Summary of COMA++ architecture

Test schema	
Schema type	XSD, XDR
#of Schemas/ match tasks	7/16
Min/Max/Avg schema size	34/26,000/-
Matching Algorithm	
Linguistic	Dictionary, string matching
Structure	leaves
Match Combination	-
Abbreviation Processing	**Manual**
Evaluation Strategy	
Quality Measure	Precision/Recall/F
Subjectivity	1 user

5.2. CUPID

Cupid discovers mappings between schema elements based on their names, data types, constraints, and schema structure, using a broader set of techniques than prior approaches. Cupid integrates linguistics and structural matching, context-dependent matching of shared types, and a bias toward leaf structure. Cupid has the following properties by Madhavan (2001).

- It includes automated linguistic-based matching.

- It is both element-based and structure-based.

- It is biased toward similarity of atomic elements (i.e. leaves), where much schema semantics is captured.

- It exploits internal structure, but is not overly misled by variations in that structure.

- It exploits keys, referential constraints and views.

- It makes context-dependent matches of a shared type definition that is used in several larger structures.

- It generates 1:1 or 1:n mappings, although Cupid is an artefact of final stage of the algorithm and could be adjusted if desired.

Linguistic matching of Cupid include normalization, categorization, and comparison. Normalization process aims to solve the uncertainty of schema

element due to the abbreviation, acronyms, punctuation, etc. Cupid normalizes schema element by following the process of tokenization, expansion, elimination, and tagging. Categorization makes a group of elements that can be identified by a set of keywords, which are derived from concepts, data types, and element names. The reason of doing this process is to decrease the number of element-to-element matching. To decide the belonged group, a name similarity measure is used. Comparison phase is to calculate the similarity between two schema elements using linguistic-based similarity measure.

Structural matching of Cupid mainly depends on tree matching algorithms. The details of tree matching algorithms can be found in figure below.

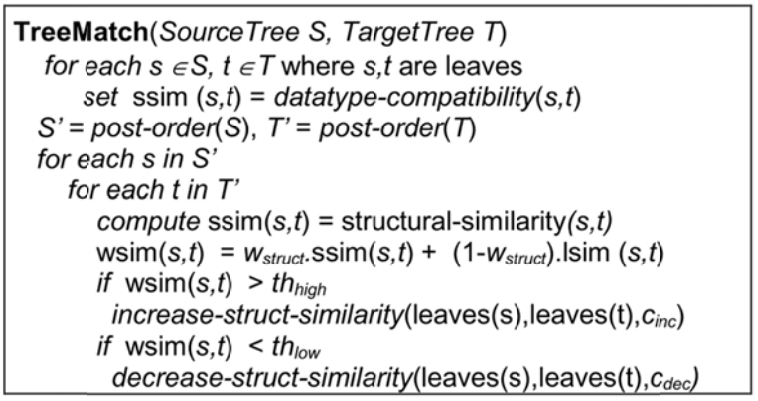


Figure 7. The Tree Matching Algorithm of Cupid (Madhavan 2001)

The intuitive motivation of tree matching algorithm is

(a) Atomic elements (leaves) in the two trees are similar if they are individually (linguistic and data type) similar, and if elements in their respective vicinities

(ancestors and siblings) are similar.

(b) Two non-leaf elements are similar if they are linguistically similar, and the subtrees rooted at the two elements are similar.

(c) Two non-leaf schema elements are structurally similar if their leaf sets are highly similar, even if their immediate children are not. This is because the leaves represent the atomic data that the schema ultimately describes.

The summary of Cupid's feature is summarized in table below.

Table 2. Summary of Cupid architecture

Test schema	
Schema type	XDR
#of Schemas/ match tasks	2/1
Min/Max/Avg schema size	40/54/47
Matching Algorithm	
Linguistic	String matching
Structure	neighbors
Match Combination	-

Abbreviation Processing	**Manual**
Evaluation Strategy	
Quality Measure	None
Subjectivity	1 user

5.3. IMAP

iMAP system semi-automatically discovers both 1-1 and complex matches. iMAP reformulates schema matching as a search in a large or infinite matching space. To search effectively, it employs a set of searchers, each discovering specific types of complex matches. To further improve matching accuracy, iMAP exploits a variety of domain knowledge, including past complex matches, domain integrity constraints, and overlap data (Dhamankar et al. 2004).

One of distinct features of iMAP is to exploit domain knowledge. Several recent works (Doan et al. 2001, Do et al. 2002, Madhavan et al. 2003) have noted the benefits of exploiting domain knowledge for schema matching. Intuitively, domain knowledge are useful for tuning some possible matches (Dhamankar et al. 2004).

The iMAP architecture is shown in figure below. It consists of three main modules: match generator, similarity estimator, and match selector.

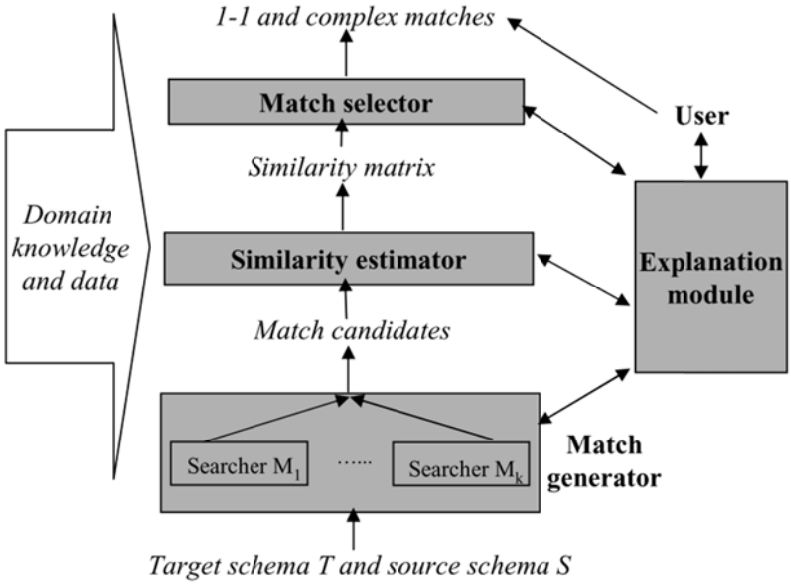


Figure 8. The iMAP architecture (Dhamankar 2004)

The match generator takes as input two schemas S and T. For each attribute t of T, it generates a set of match candidates and the generation is directed by a set of search modules. The similarity estimator then calculates for each match candidate a score which directs the candidate's similarity to attribute t. Thus, the output of this module is a matrix that stocks the similarity score of <target attribute, match candidate> pairs. Finally, the match selector observes the similarity matrix and outputs the best matches for the attributes of T. During the entire matching process, the above three modules also exploit domain knowledge and data to maximize matching accuracy, and interact with an explanation module to generate explanations for matches.

iMAP uses two main techniques to search the space effectively. First, it

adopts a set of specialized searchers that explore meaningful parts of the matching space. Second, it makes use of various types of domain knowledge to guide the search and the evaluation where possible (this means iMAP is not fully automatic approach). The main points of iMAP are summarized in table below.

Table 3. Summary of iMAP architecture

Test schema	
Schema type	DTD, HTML
#of Schemas/ match tasks	24/20
Min/Max/Avg schema size	14/66/-
Matching Algorithm	
Linguistic	String matching
Structure	N/A
Match Combination	-
Abbreviation Processing	**Manual**
Evaluation Strategy	
Quality Measure	Recall

Subjectivity	1 user

5.4. SEMINT

SEMINT[8] (Li et al. 94, Li and Clifton 2000-1, Li and Clifton 2000-2) represent a hybrid approach exploiting both schema and instance information to find semantic correspondence between attributes from relational schemas. Constraints-based matching technique is also adopted. SEMantic INTgreator (SEMINT) is a tool based on neural networks to aid in finding attribute correspondence between different and heterogeneous databases (see below figure). It supports access to various types of database systems and early version of hybrid matching approach.

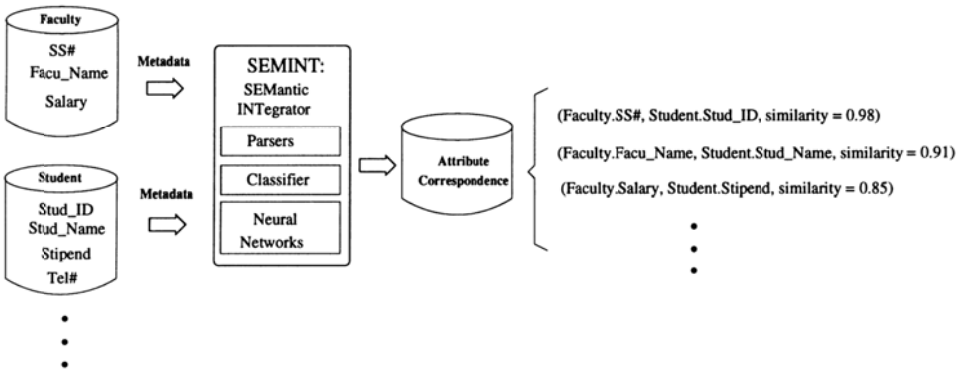


Figure 9. Overview of semantic integration in SEMINT

In the example in Li and Clifton (2000-1), they describe the situation of

[8] The objective of SEMINT is slightly different from other architectures. Most of schema matching architecture aims to find correspondence between schema elements. However, SEMINT is tool for integration of heterogeneous database, not matching.

integration between Faculty and Student databases. INTegrator (SEMINT) first uses DBMS specific parsers to extract metadata from these two databases. The metadata forms patterns labelling the attributes in the Faculty and Student databases. These attribute patterns are used as training data for neural networks to recognize these patterns. The trained neural network can then identify corresponding attributes, based on the metadata of attributes in, Faculty and Student databases and point out their similarity. In SEMINT (see Figure above) the metadata extraction is automated; and how to match corresponding attributes and determine their similarity is `learned' during the training process directly from the metadata. The features of SEMINT is summarized in table below.

Table 4. Summary of SEMINT architecture

Test schema	
Schema type	Relational
#of Schemas/ match tasks	10/5
Min/Max/Avg schema size	6/260/57
Matching Algorithm	
Linguistic	String matching
Structure	N/A

Match Combination	-
Abbreviation Processing	-
Evaluation Strategy	
Quality Measure	Precision/ Recall
Subjectivity	1 user

5.5. Similarity Flooding

Similarity Flooding (Melnik et al. 2002) transforms schemas into labelled graphs and user fix-point computation to finding correspondence. To explain the details of Similarity Flooding, Melnik et al. (2002) shows the brief example of schema matching. Consider schemas S_1 and S_2 depicted in Figure below.

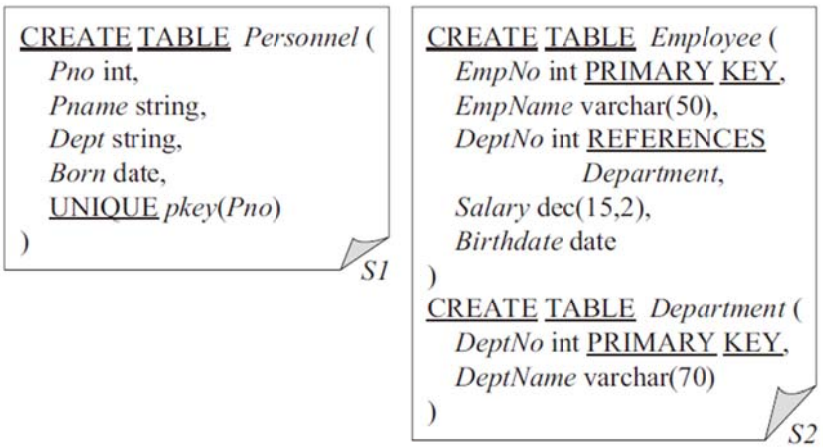


Figure 10. Example of matching two relational schemas: Personnel and Employee-Department

The elements of S_1 and S_2 are tables and columns. Assume for now that our goal is to obtain exactly one matching element for every element in S_1. A part of the matching result could be, for example, the correspondence of column Personnel/Pname to column Employee/EmpName. A sequence of steps that permits us to decide the correspondences between tables and columns in S_1 and S_2 can be expressed as the following script:

1. G_1 = SQL2Graph(S_1); G_2 = SQL2Graph(S_2);

2. initialMap = StringMatch(G_1, G_2);

3. product = SFJoin(G_1, G_2, initialMap);

4. result = SelectThreshold(product);

As a first step, schemas are transformed into two graph types. Second, an initial mapping between G_1 and G_2 is obtained using StringMatch operation. In third step, operator SFJoin is applied to produce a refined mapping called product between G_1 and G_2. Finally, operator SelectThreshold selects a subset of node pairs in product that corresponds to the 'most plausible' matching entries.

One of their contributions is the novelty of quality metric. They firstly introduce the concepts of overall (sometimes called accuracy[9]). Overall metric is designed for capturing the user effort needed to transform a match result obtained automatically into the intended result (See formula and figure below).

$$Overall = Recall*(2-1/Precision)$$

[9] Some researchers use term 'accuracy' as a meaning of 'precision'.

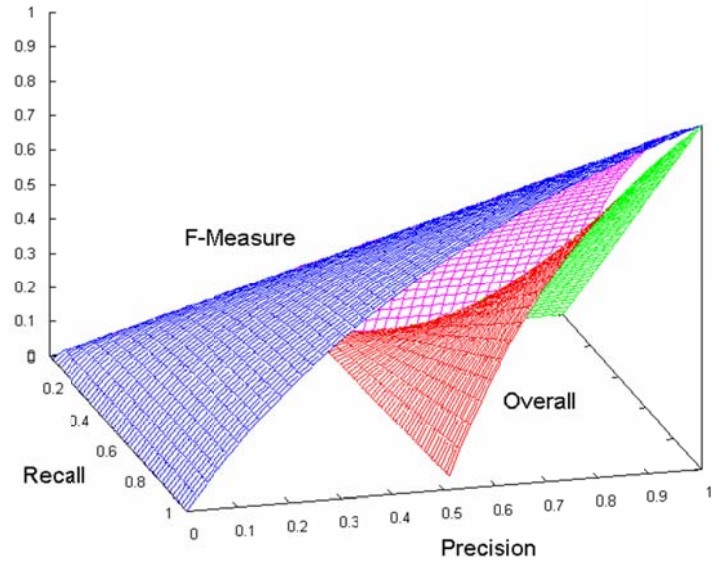


Figure 11. F-measure and Overall as functions of Precision and Recall (Do 2005)

In his dissertation, Do Hong Hai (2005) explains the meaning of overall:

"*To compare the behavior of Fmeasure and Overall, Figure 10.3 shows them as functions of Precision and Recall, respectively. Apparently, Fmeasure is much more optimistic than Overall. For the same Precision and Recall values, Fmeasure is much higher than Overall. On the other side, Overall is more sensitive to Precision than to Recall. In particular, Overall quickly degrades to 0 when Precision decreases to 0.5. If the number of the false positives exceeds the number of the true positives, i.e., Precision<0.5, Overall can have arbitrary negative values, while the minimum value of the other measures is 0. Both Fmeasure and Overall reach their highest value 1.0 with Precision=Recall=1.0. In all other cases, while the value of*

Fmeasure is within the range determined by Precision and Recall, Overall is smaller than both Precision and Recall."

The summary of Similarity Flooding architecture is presented in table below.

Table 5. Summary of Similarity Flooding architecture

Test schema	
Schema type	XSD
#of Schemas/ match tasks	18/9
Min/Max/Avg schema size	5/22/12
Matching Algorithm	
Linguistic	Affix/suffix, dictionary
Structure	Parent/ children/ leaves
Match Combination	Max
Abbreviation Processing	**Manual**
Evaluation Strategy	
Quality Measure	Overall

Subjectivity	7 user

5.6. Overall Comparison of Representative Prototype and

We reviewed representative schema matching architectures (COMA++, CUPID, iMAP, SEMINT, and Similarity Flooding). In this section, the comparison among them is analysed for this doctoral dissertation. An overall comparison is summarized in table below.

Table 6. Characteristics of representative schema matching prototype

	COMA++	CUPID	IMAP	SEMINT	SF
Test schema					
Schema type	XSD, XDR	XDR	DTD, HTML	Relational	XSD
#of Schemas/ match tasks	7/16	2/1	24/20	10/5	18/9
Min/Max/Avg schema size	34/26,000/-	40/54/47	14/66/-	6/260/57	5/22/12
Matching Algorithm					
Linguistic	Dictionary, string matching	String matching	String matching	-	Affix/suffix, dictionary
Structure	leaves	neighbors			Parent/ children/ leaves

Match Combination	-	-	-	-	Max
Abbreviation Processing	**Manual**	-	**Manual**	-	**Manual**
Evaluation Strategy					
Quality Measure	Precision/ Recall/ F	none	Recall	Precision / Recall	
Subjectivity	1 user	1 user	1 user	1 user	7 user

- Schema Type: Most recent architectures use their input data as XML schema. XSD and XDR are representative XML schema file format.
- Size of Test Schema: Architectures reviewed use the test schema for evaluation of their approach. Each architecture uses multiple schema pairs as their test schema and the size of test schema varies. It is general to use sample schema whose size is less than one hundred (actually, under fifty) because of the difficulty in making mapping dictionary for evaluation. To calculate the evaluation criteria such as F-measure, we should have the set of matched pairs given by human. Due to this difficulty, test for large scale schema matching is still challenge.
- Linguistic measure: Every reviewed architecture adopts linguistic measure. String matching measure (non-semantic measure) is most frequently adopted because most string representing schema element is too modified (i.e., abbreviation) to find element name in auxiliary information such as

WordNet and dictionary. In the case of ontology matching, various semantic measures such as node-based and edge-based similarity tend to be critical measures. To utilize semantic similarity measure in schema matching case, normalization process including tokenization, extension should be performed to identify the real meaning of modified element name.

- Abbreviation processing: It is very conventional in database modelling to naming the name of schema element using abbreviation. Most of reviewed prototypes need to human endeavour to solve the abbreviation. They use human-made abbreviation dictionary before performing schema matching task.
- Quality measure: Precision, Recall, and F-measure are most commonly used quality measure in schema matching research. Similarity Flooding is the only architecture that does not use these traditional quality measures. They suggest and use ‘overall’ measure for the first time.

To briefly comparing this dissertation and reviewed approach, the features of my dissertation approach that will be scrutinized in next part are summarized in table below.

Table 7. Properties of Schema Matching Approach in Dissertation

	Proposed Matching	**Note**
Test schema		
Schema type	XML schema	XML schema as a canonical form of schema data type
#of Schemas/ match tasks	About 100hundred	- About 50 schemas (University, car sales, auction, etc) / average size: 25 /from other schema matching researchers - 5 schemas (Purchase order)/ average size: 20/ from COMA++ - Not test the large scale schema
Min/Max/Avg schema size	-	
Matching Algorithm		
Linguistic	Hybrid	- Semantic matching : path similarity, Wu-palmer measure, Lin measure, Resnik measure (Auxiliary data: WordNet, Rodget's theasauraus, Brown Corpus). - Non-semantic matching (Syntactic measure): Levenshtein distance, Jaro similarity, Jaro-winkler similarity, N-gram distance. - Cross Similarity Vector approach using similarity measure (linguistic-structural matching).
Structure		
Match Combination	Learning-based Classification	- KNN/SVM/Decision Tree
Abbreviation Processing	No, Fully automated	- For matching schemas from generalized domain
Evaluation Strategy		
Quality Measure	Precision/ Recall/ F	-
Subjectivity	1 user	Implementing manual matching solution.

PART3. CROSS SIMILARITY VECTOR APPRAOCH

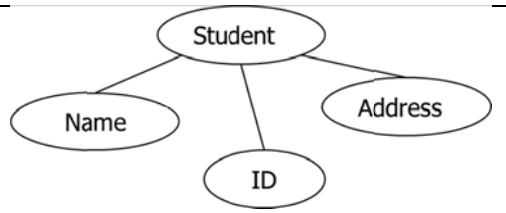
CHAPTER6. Motivation for Cross Similarity Vector Approach

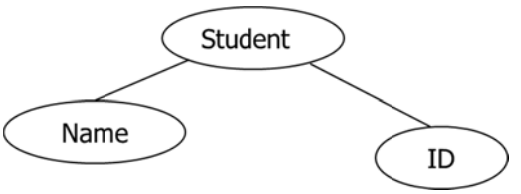
6.1. Element and structural similarity between schema elements

Many algorithms for calculating similarity between schema elements have been suggested. As presented in Chapter 4, generally, individual matchers can be classified into element-based and structural similarity. Element based matching finds the correspondence based on the name of elements or constraints. Structural matching mainly uses the topological approach (Melnik et al. 2002, Bertino et al. 2004, Fan et al. 2010). The common utilization of schema's structural aspect is to consider the parents, children, or the leaves of each schema element.

Most of structural measures only reflect structural aspect such as topological position, the number of neighbours, etc. Structural approach plays an important role to filtering the result of element based matching. Two elements must be un-matched pair if the hierarchical order of is not match though they have a high name similarity value. However, here's the sceptical aspect of structural matching. See the table below. To cope with this critical situation, a novel approach to calculate semantic structure similarity is proposed.

Example	Matching Result between two "Student" in source and target

<table>
<tr>
<td> 

(source)</td>
<td>Existing Structural Matching Measure
: NOT Match because two entity has different morphology considering the number of child (linked leaf).</td>
</tr>
<tr>
<td> 

(target)</td>
<td><u>Proposed Measure</u>
: Match because two elements have same semantic relationship structure even though they have different morphology.</td>
</tr>
</table>

In this dissertation, structural approach which can reflect the context of each element is proposed. Though two elements have different number of child, parents, and ancestor, if they have same context-structure, they should have high value of similarity.

For this novel approach, we should define the concept of 'context structure' and 'context' based on Kashyap and Sheth' (1996), Goh et al.'s (1999) and Bohannon et al.'s research (2006).

Definition. The **context** is the key component in capturing the semantics related to an object's definition and its relationships to other objects (Kashyap and Sheth 1996). In this manner, the context of schema element can be described using the relation between elements and neighboring element. Context can play a critical role in capturing schema information (Bohannon and Elnahrawy 2006).

Definition. If we use context as a proxy which can reflect properties of schema element or database design (Kashyap and Sheth 1996) to find the matching between schema elements, we can call this kind of approach c**ontext structure-based approach**. That is, in this approach, semantic correspondence between schema elements is determined based on the semantic similarity between contexts of schema elements. To calculate the similarity between contexts of schema elements, explicit representation of context should be defined.

In this regard, context structure-based CSV (Cross Similarity Vector) approach is proposed. In next section, the graphical description of CSV will be presented.

6.2. Graphical description of cross similarity vector approach

The figure below presents a graphical description of cross similarity vector approach. Let assume that we want to perform matching task between two schema elements and attributes.

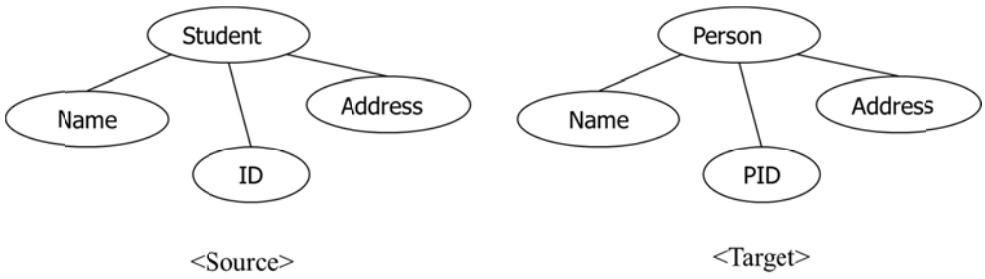


Figure 12. Example schema for cross similarity vector appraoch

To comparing 'Student' and 'Person', we should define neighbor concept vector.

Neighbor concept vector of 'Student' and 'Person' are (Name, ID, Address) and (Name, PID, Address).

Similarity Vector can be defined by conducting similarity vector operation

Original Similarity Vector of Source: (Name, ID, Address)*Student = (0.7, 0.6, 0.4)

Cross Similarity Vector of Source: (Name, ID, Address)*Person = (0.6, 0.6, 0.3)

Original Similarity Vector of Target: (Name, PID, Address)*Person = (0.6, 0.4, 0.3)

Cross Similarity Vector of Source: (Name, PID, Address)*Student = (0.7, 0.35, 0.4)

Element Similarity ES(Student, Person) = 0.88

To comparing 'Student' and 'Name', we should define neighbor concept vector.

Neighbor concept vector of 'Student' and 'Name' are (Name, ID, Address) and

(Person, PID, Address).

Original Similarity Vector of Source: (Name, ID, Address)*Student = (0.7, 0.6, 0.4)

Cross Similarity Vector of Source: (**Name**, ID, Address)***Name** = (**1**, 0.5, 0.3)

Original Similarity Vector of Target: (Person, PID, Address)*Name= (0.6, 0.5, 0.3)

Cross Similarity Vector of Source: (**Person**, PID, Address)***Student** = (**0.9**, 0.35, 0.4)

Element Similarity ES(Student, Person) = 0.62

CHAPTER7. Preliminaries for Cross Similarity Vector Approach

- Neighbor concept vector $\vec{e} = (e_1, e_2, \ldots e_n)$

  Where $e_1, e_2, \ldots e_n$ are concepts that are directly linked and parallel to concept e. Concept vector has the component with order.

- Similarity function $Sim(c_1, c_2)$ which is predefined measure for calculating the similarity between given concepts. This function returns the value between zero and one.

- Similarity Vector Operation * and Similarity Vector:

  $\vec{e} * e' = (e_1, e_2, \ldots e_n) * e' = (Sim(e_1, e'), Sim(e_2, e'), \ldots Sim(e_n, E')) = \overrightarrow{SV}_{ee'}$

- Original Similarity Vector of Source $\overrightarrow{SV}_{ee}$ = $\vec{e}$ *e= (e_1, e_2, ...e_n)*e=(Sim(e_1,e), Sim(e_2, e), ...Sim(e_n, e))

- Cross Similarity Vector of Source = $\overrightarrow{SV}_{ee'}$ = $\vec{e}$ *e'= (e_1, e_2, ...e_n)*e'=(Sim(e_1,e'), Sim(e_2, e'), ...Sim(e_n, e'))

- Original Similarity Vector of Target $\overrightarrow{SV}_{e'e'}$ = $\overrightarrow{e'}$ *e'= (e_1', e_2', ...e_m')*e'=(Sim(e_1',e'), Sim(e_2', e'), ...Sim(e_m', e))

- Cross Similarity Vector of Target $\overrightarrow{SV}_{e'e}$ = $\overrightarrow{e'}$ *e= (e_1', e_2', ...e_m')*e=(Sim(e_1',e), Sim(e_2', e), ...Sim(e_m', e))

  (note that the number of neighbor concepts, n and m, may be different)

- There are two option to calculate Element Similarity according to the way of compute the similarity between two vectors which have same dimension.

  - Option 1. ES (*e, e'*) = Average [Corr($\overrightarrow{SV}_{ee}$, $\overrightarrow{SV}_{ee'}$) , Corr ($\overrightarrow{SV}_{e'e'}$, $\overrightarrow{SV}_{e'e}$)] which is the average of Pearson correlation between two pairs of vectors.

  - Option 2. ES (*e, e'*) = Average [CosSim($\overrightarrow{SV}_{ee}$, $\overrightarrow{SV}_{ee'}$) , CosSim ($\overrightarrow{SV}_{e'e'}$, $\overrightarrow{SV}_{e'e}$)] which is the average of cosine similarity between two pairs of vectors.

- Comparison between cosine similarity and Pearson correlation.

- Cosine Similarity

$$\mathrm{CosSim}(\mathrm{x},\mathrm{y}) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}} = \frac{<x,y>}{||x||||y||},$$

where <x, y> is the dot product.

- Pearson Correlation

$$\mathrm{Corr}(\mathrm{x},\mathrm{y}) = \frac{\sum_i (x_i - \bar{x})\,(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2}\sqrt{\sum (y_i - \bar{y})^2}}$$

$$= \frac{< x - \bar{x}, y - \bar{y} >}{\|x - \bar{x}\|\|y - \bar{y}\|}$$

$$= \mathrm{CosSim}(x - \bar{x}, y - \bar{y})$$

Correlation is the cosine similarity between centered versions of x and y, again bounded between -1 and 1

CHAPTER8. Experimental Result of Cross Similarity Vector Approach

8.1. Experimental evaluation of CSV (Cross Similarity Vector) approach

In this section, the validity of CSV approach will be evaluated by comparing CSV result, name similarity, and existing structural measure according to various

topological structures of schemas to be matched.

8.1.1. CSV evaluation - Two schema which have same structure

First, the case that two schemas has same structure is investigated. The structure of source and target schema is depicted in the figure below. We use the jaro-winkler measure as Similarity function $Sim(c_1, c_2)$ for computing the components of cross similarity vectors.

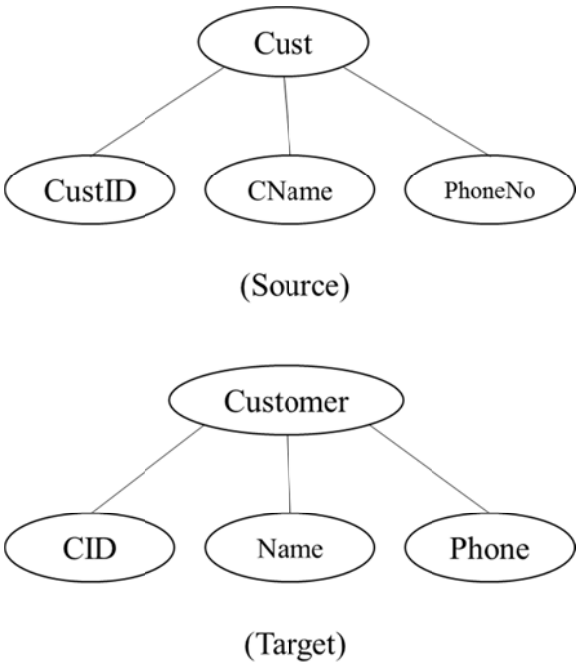


Figure 13. Two schema with same topological structure

To certify the applicability of CSV approach, we compare the matching result of name similarity, structural similarity, and CSV similarity.

Table 8. The result of CSV approach for two schema with same topological structure

Source	Target	Match	Name Similarity	Structure-based	CSV measure
cust	customer	O	0.9	Uniquely match	0.748467
cust	CID	X	0	Not match	-0.1
cust	name	X	0	Not match	-0.3
cust	phone	X	0	Not match	0.143
CustID	customer	X	0.625	Not match	0.01
CustID	CID	O	0.5	Match but Not Unique	0.61
CustID	name	X	0	Match but Not Unique	-0.13
CustID	phone	X	0	Match but Not Unique	-0.21
cname	customer	X	0.65833333	Not match	0.224
cname	CID	X	0	Match but Not Unique	0
cname	name	O	0.93333333	Match but Not Unique	0.8
cname	phone	X	0.46666666	Match but Not Unique	0.3

phoneNo	customer	X	0.51190476	Not match	0.1
phoneNo	CID	X	0	Match but Not Unique	-0.25
phoneNo	name	X	0.46428571	Match but Not Unique	0.32
phoneNo	phone	O	0.94285714	Match but Not Unique	0.99

8.1.2. CSV evaluation - Two schema which have different structure

In this section, the schemas which have different topological structures is tested for evaluation. The distorted schema from prior section is used for this experiment (see figure below).

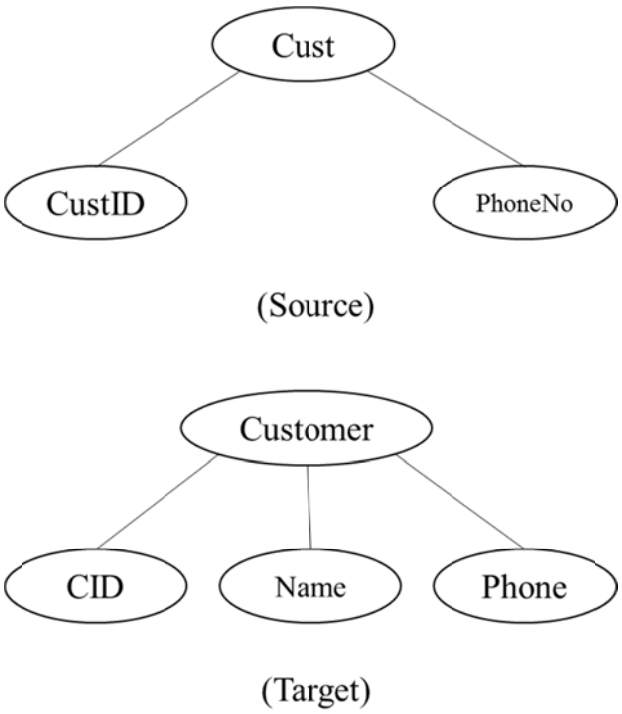


Figure 14. Two schema with different topological structure

The entity 'CName' is eliminated from source schema comparing to prior experiment and this causes the change in terms of structural property of schema. Structural measure based on the hierarchical linkage shows very low discriminant in finding matched element pairs. The details are summarized in table below.

Source	Target	Match	Name Similarity	Structure-based	CSV measure
cust	customer	O	0.9	Not match	**0.96**
cust	CID	X	0	Not match	0.21
cust	name	X	0	Not match	-0.213

cust	phone	X	0	Not match	-0.22
CustID	customer	X	0.625	Not match	0.32
CustID	CID	O	0.5	Match but Not Unique	**0.72**
CustID	name	X	0	Match but Not Unique	-0.6
CustID	phone	X	0	Match but Not Unique	-0.21
phoneNo	customer	X	0.51190476	Not match	0.1
phoneNo	CID	X	0	Match but Not Unique	0.209
phoneNo	name	X	0.46428571	Match but Not Unique	0.38
phoneNo	phone	O	0.94285714	Match but Not Unique	**0.99**

8.1.3. Overall matching performance of CSV

To prove the efficiency of CSV approach, we compare the matching performance between two cases: with only string-based measure vs. with string-based measure and its CSV measure. As a string-based measure, jaro-winkler measure are used. Matching strategy is adopted from Part 4 of this dissertation. We conduct 20 times

of training and testing for each case and compare the average F-measure. The difference of F-measure for each case is statistically tested based on t-test. Figure below shows the distribution of f-measure for each matching experiment.

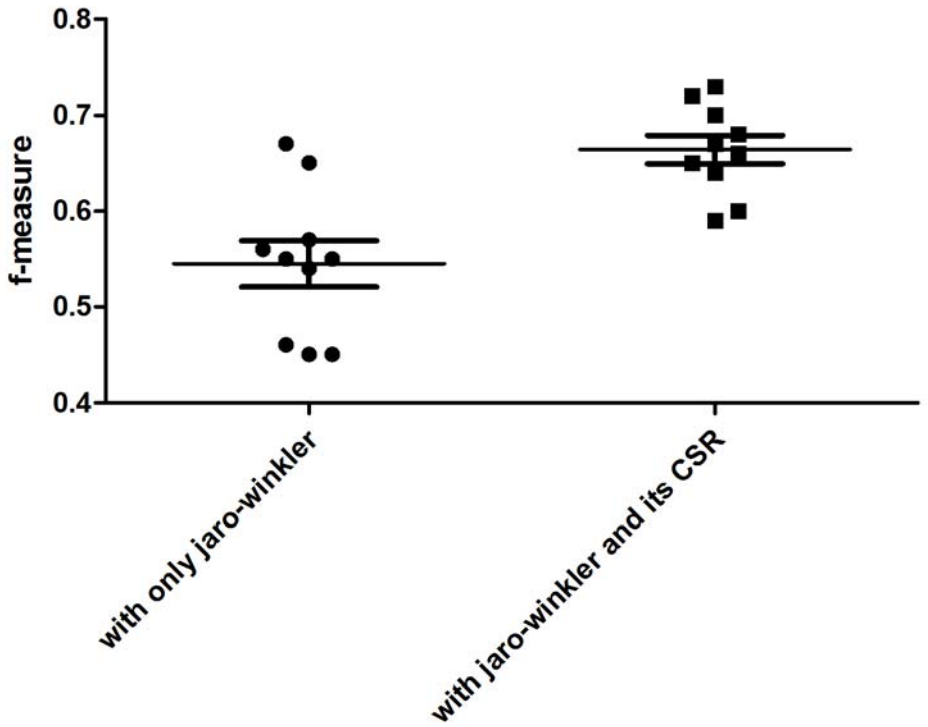


Figure 15. The distribution of f-measure – with only jaro-winkler vs. with jaro-winkler and its CSR.

CSR measure can make statistical different when it is used with string-based matcher. Table below shows the summary of statistical test.

Table 9. Summary of statistical test for the effect of CSR

Unpaired t test	
P value	0.0005
P value summary	***
Are means signif. different? (P < 0.05)	Yes
One- or two-tailed P value?	Two-tailed
t, df	t=4.199 df=18

8.2. Discussion for CSV based on experiment result

CSV measure returns significantly better performance than existing structural measure which reflect the morphological property of schema element. CSV is dimension free. Though each source and target element has different number of neighbour elements, the metric does not be affected from the situation (the number of neighbour element indicates the dimension of cross similarity vector). By reflecting the context structure of schema element, the effect of simple structural change such as the change of ancestor, siblings' number is dramatically diminished.

There are two option to calculate the CSV measure as stated in preliminary chapter. We adopt the Pearson correlation-based measure to calculate the similarity between cross similarity vector. Comparing to the cosine similarity measure, Pearson measure show better performance.

The jaro-winkler measure is used for similarity function $Sim(c_1, c_2)$ to compute the components of cross similarity vectors in the experiment. Other alternatives for similarity function such as edit distance measure, N-gram measure also show very robust result. We do not use the semantic measure for similarity function because most of schema elements should be manually processed to extract the abbreviations to apply semantic measure.

PART4. LERANING-BASED MATCHER COMBINATION STRATEGY

CHAPTER9. Combinational and Hybrid Matcher Scenario

9.1. Schema Matching – Multiple Matcher Approach

Single matcher generally focuses on solving specific matching situation. This means each single matcher utilizes different information from different matching scenario and individual matcher tend to have applicability and value for a specific given matching task. To implement schema matching for more general matching problem, more individual matchers should be integrated and composed according to given matching situation. Therefore, a matcher that uses just one matching approach is unlikely to accomplish as many good match candidates as one that combines several approaches (Rahm and Bernstein 2001).

There're two way of combining different matcher: hybrid and composite match approach. The hybrid approach utilize different match criteria within a single algorithm. By contrast, a composite match approach combines the results of several independently executed match algorithms, which can be simple (based on a single match criterion) or hybrid (Do and Rahm 2002). This enables a high flexibility, as there is the potential for selecting and combining the match algorithms to be applied based on a condition of given matching task.

Early research of composite matcher approach can be find in few research

(Doan et al. 2001, Embley et al. 2002, Doan et al. 2002). These studies adopt machine learning approach to implement composite matcher but they fail to fully utilize the strengths of composite matcher such as its flexibility and applicability to various matching scenario. Comparing theses research, COMA/COMA++ shows better performance and flexibility by using advanced matcher combination. COMA++ execute whole individual matchers and the result is aggregated into similarity cube. Though COMA++ does not show the scientific method to decide weighting parameters and matching threshold, it is high level of matching performance with abbreviation processing and user feedback. Most schema matching research adopt the result of COMA++ as their benchmark. Cupid represents a sophisticated hybrid match approach combining a name and structural match algorithm, which derives the similarity of elements based on the similarity of their components (the name and data type similarities in leaf level). In a comparative evaluation Cupid was generally more effective than earlier match prototypes such as Dike (Palopoli et al. 2000) and Momis (Bergamaschi et al.).

In past research on combinational matching approach, some weaknesses can be found. Most of parameter tend to be determined based on iterative approach. Due to the lack of rigorous and scientific justification, criteria for deciding parameter seem very arbitrary. In addition, matching result strongly depends on a human intervention in the case of semi-automatic schema matching. Most of schema elements are typed in abbreviation form according to the habit of database modeller. This point is one of the most difficult problem to make schema matching

fully automatic.

In this part, I suggest fully automated schema matching using multiple matchers without any human intervention. By transforming the multiple matcher combination into learning-based classification problem, the complexity of combination problem decrease as the number of matcher increase. The matchers used for matching algorithm are selected based on the learning algorithm and utilized for features for classification. As matcher[10] returns real value between 1 and 0, I can enjoy the pre-normalized property of each feature. Machine learning-based classification tend to perform better as features has same scale. In next section, I revisit the formal representation of schema matching for transformation of matching into classification problem.

9.2. Formal Representation of Schema Matching Task

Schema matching has been defined throughout many past research. In the early research on schema matching, it is defined as the process of matching vertices of the source schema graph with vertices of the target schema graph (Milo and Zohar 1998). Giunchiglia et al. (2005) proposes a concept of semantic schema matching that is based on the two ideas: (1) discovering mapping by computing semantic relations such as equivalence, more general, etc; (2) determining semantic relations by analysing the meaning (concepts, not labels) which is codified in the elements

[10] Many kind of similarity measures for string comparison are adopted.

and the structures of schemas. Rahm and Bernstein (2001) defines the *Match* as a fundamental operation in manipulation of schema information. *Match* indicates the operation that takes two schemas as input and produces a mapping between elements of two schemas that correspond semantically to each other (Miller et al. 1994, Palopoli et al. 1998, Doan et al. 2000). Evermann (2009) emphasizes the importance of schema matching task as a pre-step of database integration which is increasingly important activity to ensure the continuing performance and competitive advantage of business. Under the common perspective on the importance of schema matching, *Matching* operation is the critical factor of schema matching task.

As depicted in section 3.2., Matching operation can be seen as the figure below (for readability, revisit the figure in 3.2.).

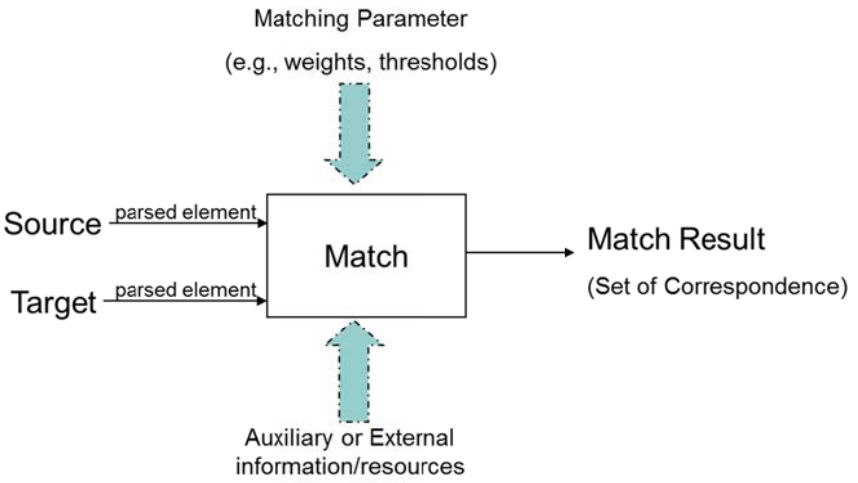


Figure 16. Forma representation of schema matching

Source and Target schema should be parsed into their schema element to find a correspondence between their elements. External information and resource such as dictionary, WordNet (Christiane 1998) might be loaded before match operation. Match operation might use single or multiple matching algorithms. In multiple matcher approach, the way of combination among matchers should be determined. Most common form of combination is weighted sum of individual matcher[11]. In this case, Match operation can be expressed below.

$$Match(e_1, e_2) = w_1 sim_1(e_1, e_2) + w_2 sim_2(e_1, e_2) + \ldots + w_n sim_n(e_1, e_2)$$

where w_n is the weight of n-th similarity measure, and e indicate the parsed schema element. Every possible pairs of schema elements will be calculated using Match operation and pairs whose value of Match operation exceed threshold are classified into match result (i.e., set of matching correspondence). In this regard, weights for individual similarity measure and threshold can be considered as the parameter for matching process.

[11] In this dissertation, individual matcher indicate a measure for calculating similarity between schema elements.

CHAPTER10. Transformation of Schema Matching Problem into Learning-based Classification

10.1. Process of Machine Learning-based Classification

Machine learning is the process of learning a set of rules from instances (examples in a training set), or creating a classifier that can be used to generalize from new instances (Kotsiantis 2007). The general process of applying supervised machine learning to real world problem is depicted in figure below.

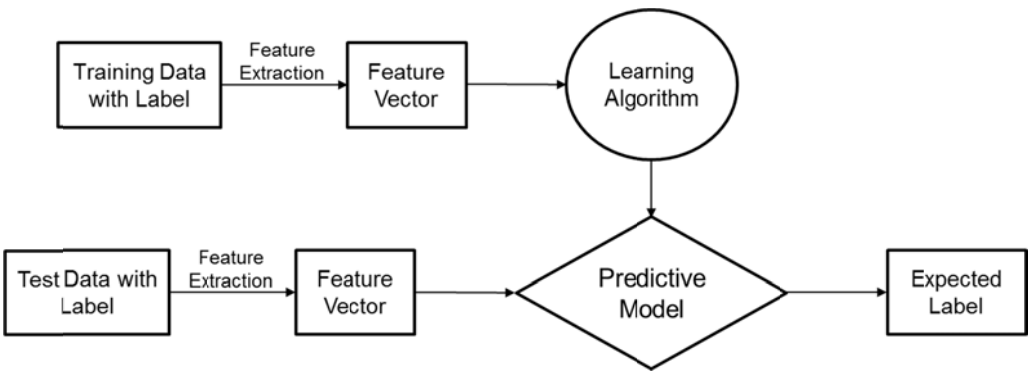


Figure 17. The process of supervised Machine Learning

In this dissertation, one of specific approach of machine learning, statistical learning-based classification will be explained as the metaphor of schema matching problem. In statistical classification, whole process and guideline might be simplified following three steps.

Step1. Observe the training set

- What makes difference between groups?

- Choose features for classification (Feature selection / extraction)

- Make N-tuples of features

Step2. Train the machine

- Using selected feature, machine can be trained.

- Select appropriate classification models/ algorithms

Step3. Machine classified the test set

- Evaluate the classification result for test data set.

Throughout the classification process, feature selection is one of most critical stage to improve the performance of classifier. Features who have higher statistical power to distinguish the training set should be extracted for test data. Generally, the more feature return higher classification performance. However, as the computation also increase as the number of feature increase, an appropriate number of feature should be determined for large-scale data set classification. Overall statistical classification process is depicted in figure below.

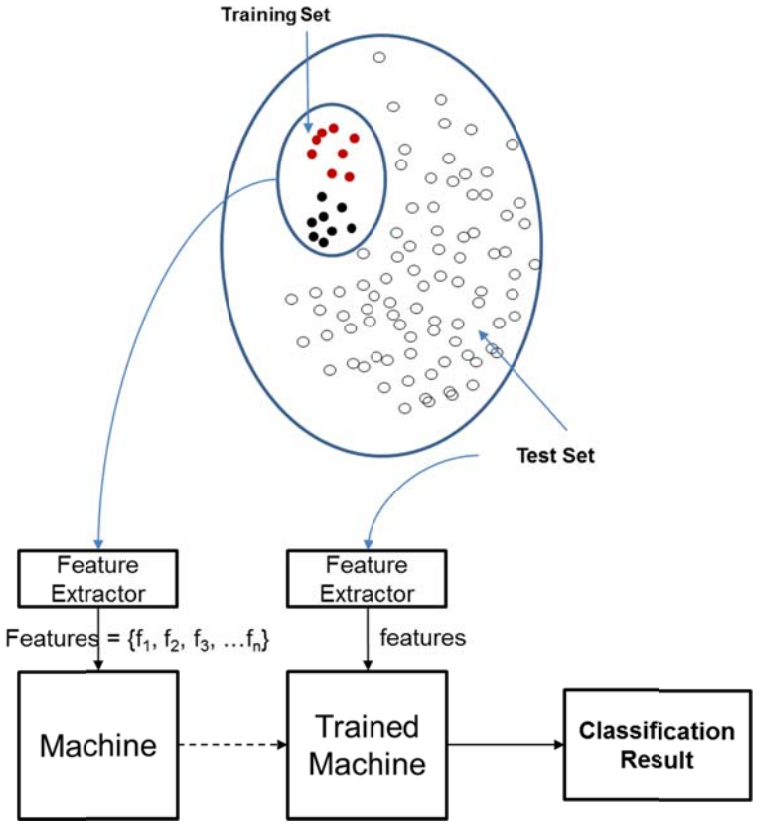


Figure 18. Overall Process of Statistical Classification

10.2. Similarities between Learning-based Classification and Schema Matching Problem

We reviewed the process of statistical classification in last section. It is not hard to find the similarity between matching and classification problem. In matching case, whether pairs of element match or not is determined by comparing threshold and the return value of Match operator. This indicate that matching problem can be seen as classification problem of which matching criteria is threshold parameter.

Individual matcher in multiple matcher problem also can be considered as feature in machine learning-based classification. Figure below show the structural similarity between schema matching problem and binary classification.

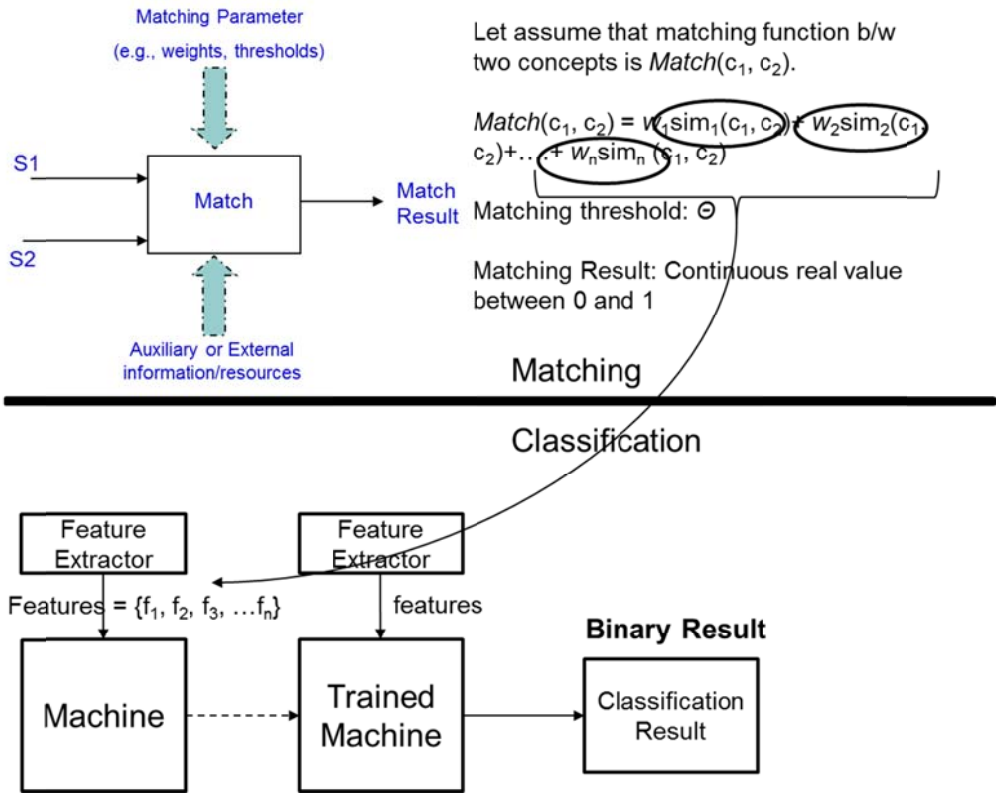


Figure 19. Comparison of Classification with Matching Problem

Based on this comparison, transformation of schema matching into classification problem will be presented in next section.

10.3. Schema Matching Problem as a Learning-based Classification

Based on prior discussion on the relationship between schema matching and classification problem, we can figure out how their transformational relationship should be.

In transformed process, each similarity measure in multiple matcher approach can be regarded as the feature that appropriately represent the property of classified group. Multiple matcher approach with n-similarity measure is transformed into the classification problem in n-dimensional feature space. Match operation is

$$Match(e_1, e_2) = w_1 sim_1(e_1, e_2) + w_2 sim_2(e_1, e_2) + \ldots + w_n sim_n(e_1, e_2)$$

and each result of match for pairs of schema elements can be expressed in the feature vector form with label[12]

$$\text{Feature vector} = (f_1, f_2, \ldots, f_n) = (sim_1(e_1, e_2), sim_2(e_1, e_2), \ldots, sim_n(e_1, e_2))$$

The details of transformation will be presented in chapter 12.

As we deal the matching problem as classification, matching process can be explained in more easy and less complex manner. In matching problem, especially in the case of linear combination approach of individual matcher, the computational complexity is proportional to the number of matcher used. If we add some similarity measure for multiple matching, the complexity to decide matching

[12] In this case, label is not expressed in formula.

parameter is increase. In contrast, insert similarity measure in classification problem means that the dimension of feature space increase. This does not mean that the complexity of classification increase.

For successful learning-based classification, it is important to select appropriate classification algorithms and apply this algorithm to classification model. In this regard, next chapter will be dedicated to explain the concept of representative classification algorithms.

CHAPTER11. Machine Learning Algorithms for Classification

11.1. K-Nearest Neighbor Algorithm

In machine learning and pattern recognition, the k-Nearest Neighbors algorithm (or k-NN for short) is a non-parametric method used for classification and regression (Altman 1992). The training sets are feature vectors in n-dimensional feature space, each with a class label, i.e., membership. The training step of the algorithm stores the feature vectors and class labels of the training samples. One training sample is considered as one feature vector. In the classification step, k is a user-defined constant, and an unlabelled vector (a query point) is classified by assigning the label which is most frequent among the k training samples nearest to that query point. KNN has some strong consistency results. As the amount of data approaches infinity, the algorithm is guaranteed to yield an error rate no worse than twice the Bayes error rate (the minimum achievable error rate given the distribution of the data) (Cover and Hart 1967). A commonly used distance metric for continuous

variables is Euclidean distance. Euclidean distance between the point *x* and *u* in the n-dimensional vector space is

$$\mathrm{d}(\mathrm{x},\mathrm{u}) = \sqrt{\sum_{i=1}^{n}(x_i - u_i)^2}.$$

The basic KNN classification uses uniform weights: that means the value assigned to a query point is computed from a simple majority vote of the nearest neighbor. Under some circumstances, it is better to weight the neighbors such that nearer neighbors contribute more to the fit. In this case, the invers of distance become the weights.

KNN has many strengths; easy to understand and to implementation using programming language such as Java, Python, etc as well as machine learning package software. In addition, it is easy to hand the missing value by restricting distance calculation to subspace. The asymptotic misclassification rate (as the number of data points n goes to infinite) is bounded above by twice the Bayes error rate (Duda and Hart 2002).

On the other hand, KNN algorithm tends to be affected by local structure. To calculate the distance in n-dimensional space, the computational expense is relative higher than other algorithms. It also needs large memory requirement in large test sample case as all of the sample data should be hold while machine classify test data.

Determining k is one of the most important task and the performance of

KNN is likely to depend on appropriate k. If we choose smaller k, the result have higher variance with less stable. In larger k scenario, results have higher bias with less precision. Proper choice of k depends on the data. k is usually determined through iterative training and evaluation. The general thumbs-up rule is to assign the square-root of the number of features to k.

11.2. Supported Vector Machine

In machine learning, support vector machines (SVM) are supervised learning methods with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. Given a set of training sets, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new samples into one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in n-dimensional feature space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. If we fail to make a clear gap, transformation can be applied to make enough margin between hyper plane and points in category. New examples are then plotted into that same space and predicted to belong to a category based on which side of the gap they fall on.[13]

---

[13] Modified from Wikipedia , support vector machine

http://en.wikipedia.org/wiki/Support_vector_machine

The advantages of support vector machines are (Pedregosa et al. 2011):

- Effective in high dimensional spaces.
- Still effective in cases where number of dimensions is greater than the number of samples.
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- Versatile: different *Kernel functions* can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

The disadvantages of support vector machines include (Pedregosa et al. 2011):

- If the number of features is much greater than the number of samples, the method is likely to give poor performances.
- SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.

Representative SVM approaches are linear and non-linear SVM. Linear SVM finds Maximum-margin hyper-plane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors. Its concept is depicted in figure below.

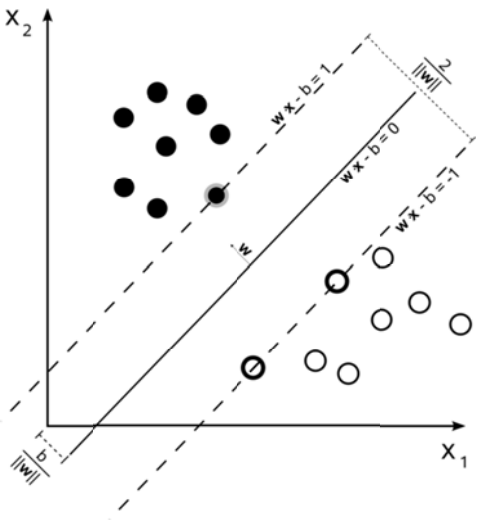


Figure 20. Linear SVM

Linear SVM can be seen as following optimization problem:

Given some training data D, a set of n-points of the form in p-dimensional space

$$D = \{(x_i, y_i) \mid x_i \in R^p, y_i \in \{-1, 1\}\} \text{ for } i=1 \sim n.$$

$$w \cdot x - b = 0,$$

Minimize $||w||$ subject to $y_i(w \cdot x_i - b) \geq 1$

Non-linear SVM is very useful when the linear SVM fail to find hyper-plane. Non-linear SVM uses transformation function to make feature space linearly separable. Figure below show the situation where the non-linear SVM should be applied.

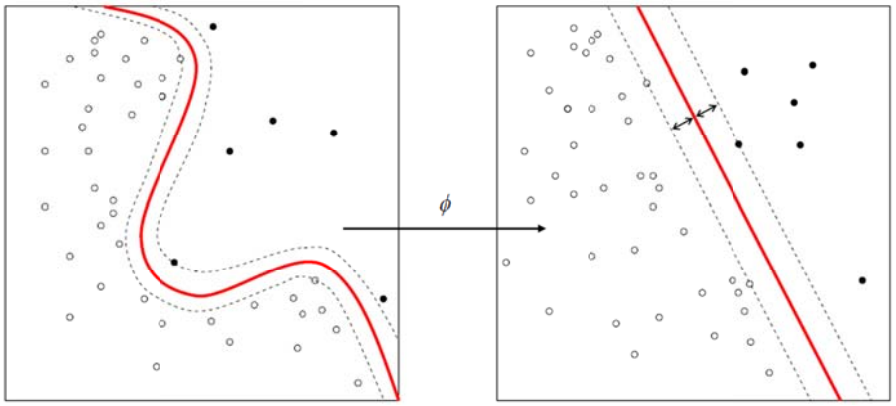


Figure 21. Non-linear SVM with transformation.

We call this transformation function 'kernel function.' Polinomial, Gaussian radial basis function, and hyperbolic tangent are most frequently used kernel function in non-linear SVM.

11.3. Decision Tree

Decision tree learning uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. It is one of the predictive modelling approaches used in statistics, data mining and machine learning. More descriptive names for such tree models are classification trees or regression trees. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.[14]

ID3 (Iterative Dichotomiser 3) was established by Ross Quinlan (1993).

[14] http://en.wikipedia.org/wiki/Decision_tree_learning

The algorithm makes a multiway tree, finding for each node the categorical feature that will yield the largest information gain for categorical targets. Trees are grown to their maximum size and then a pruning step is usually applied to increase the ability of the tree to generalize to test data.

C4.5 is the successor to ID3 and removed the limit that features must be categorical by dynamically defining a discrete attribute (based on numerical variables) that partitions the continuous attribute value into a separate set of intervals. C4.5 changes the trained trees (i.e. the output of the ID3 algorithm) into sets of if-then rules. These accuracy of each rule is then evaluated to determine the order in which they should be applied. Pruning is done by removing a rule's precondition if the accuracy of the rule improves without it.

C5.0 is Quinlan's latest version release under a proprietary license. It uses less memory and builds smaller rulesets than C4.5 while being more accurate.

CART (Classification and Regression Trees) is very similar to C4.5, but it differs in that it supports numerical target variables (regression) and does not compute rule sets. CART constructs binary trees using the feature and threshold that yield the largest information gain at each node (Pedregosa et al. 2011). Information gain is based on the concept of entropy from information theory.

$$I_E(f) = -\sum_{i=1}^{m} f_i log_2 f_i$$

Decision tree has been used for many advantage since it has been introduced. It is

very simple to understand and interpret due to the intuitive structure and requires little data preparation. Also it is able to hand both numerical and categorical data and possible to validate a model using statistical tests. It usually shows a robust performance and performs well with large data sets. In contrast, the problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts (Hyafil and Rivest 1976, Murthy 1998). Decision-tree learners can create over-complex trees that do not generalise well from the training data (known as over-fitting) (Hand 2001)

CHAPTER12. Learning-based Matcher Combination

12.1. Supervised Learning-based Binary Classification for Schema Matching

In chapter 10, we already scrutinize the possibility of transformation of matching problem into learning-based classification. In this section, we firstly propose the multiple matcher approach for schema matching based on the concept of supervised-learning based binary classification. Overall process of schema matching based on classification can be depicted in figure below.

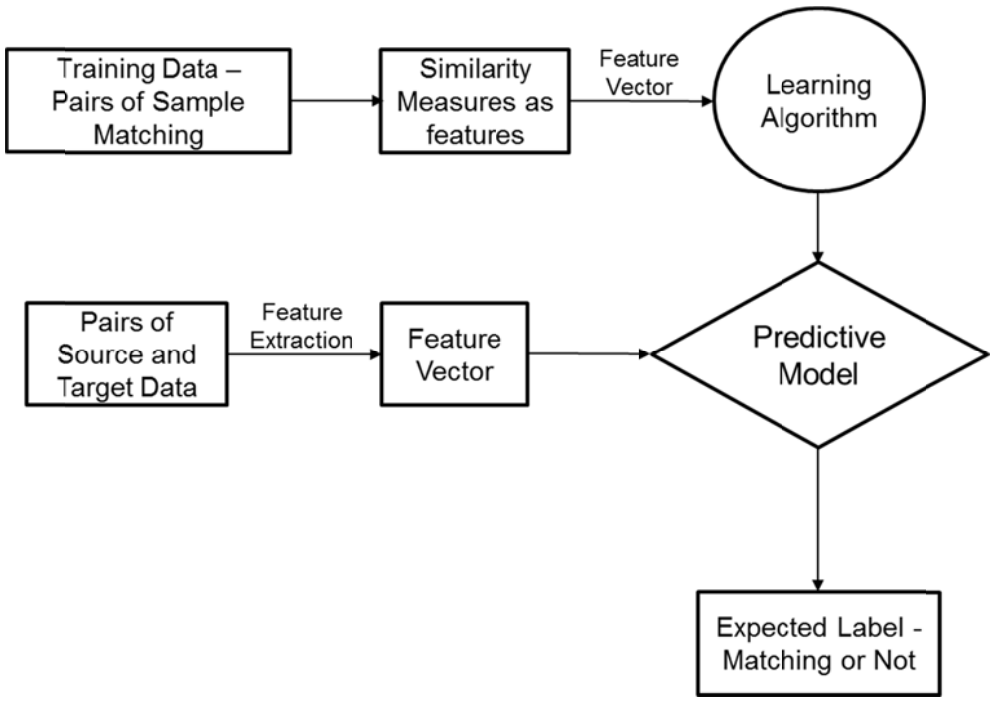


Figure 22. Schema matching based on the supervised learning-based binary classification.

As training data, the equal number of matched and un-matched pairs of schema element will be used. For making training set, small portion of test schema elements will be matched by human expert.[15] For training data, some features will be selected. In this approach, we use multiple matchers as features for training set. Similarity measures between schema elements are considered as the components

[15] In most of schema matching evaluation scenario, researchers have exact matching result to evaluate their algorithms. However, in real situation, little portion of matching between source and target element should be found to make data set for training.

representing high dimensional feature space. For a given training pairs (e1, e2) that is labelled, feature vector might be expressed in

$$(e_1, e_2) = (f_1, f_2, \ldots, f_n) = (\text{sim}_1(e_1, e_2), \text{sim}_2(e_1, e_2), \ldots, \text{sim}_n(e_1, e_2))$$

This means one sample training pairs will be plotted as one point in n-dimensional feature space. With these training feature vectors and a given classification algorithm, a classification model is defined. After machine is trained, input pairs of schema element presented in feature vector form are loaded for getting label; match or not-match.

12.2. Similarity Measures as Features in Classification

A feature selection[16] is the process of selecting a subset of relevant features for use in classification model construction. This process is very critical step for enhancing the classification performance. The main assumption of a feature selection is that data in feature space may contain many redundant or irrelevant features. By performing feature selection, training time can be reduced and the model interpretability can be improved comparing adopting all the features without selection.

In multiple matcher approach of schema matching as classifier, individual matcher (similarity measure) is considered as single feature. Prior to selecting feature, each similarity measure used for classification should be examined in

16 A feature selection is also known as variable selection, attribute selection or variable subset selection.

advance.

12.2.1. Semantic similarity measure

- **Topological Method.**

The study of similarity between concepts is meant to discover how a computational process can model the action of a human to determine the relationship between two concepts. The reason that many studies on semantic relatedness or semantic similarity use semantic networks is that the system of human knowledge can be well expressed in the form of a network. Therefore, many topological methods using human knowledge expressed in the form of a network have produced relatively good performance.

The topological method is used to calculate relatedness or similarity between concepts based on various forms of a semantic network including a hierarchical taxonomy. The nodes in a network represent concepts, and ways to measure the conceptual similarity between two nodes are also regarded as ways to determine the conceptual similarity of two words (i.e., two nodes in a network). Approaches to measuring semantic relatedness or similarity can be categorized as node-based or edge-based, which are also called the information content approach and the conceptual distance approach, respectively.

Node-based approach

The node-based approach is used to calculate similarity between concepts based on how much information the two concepts share in terms of a semantic network or taxonomy. This is why the node-based approach is also called the information content approach. The node-based approach is mostly based on Resnik's ideas, which are built on information theory (Resnik 1995). According to information theory, the information content (IC) of concept *C* can be quantified as follows:

$$IC(C) = \log^{-1}P(C),$$

where P(*C*) is the probability of encountering an instance of concept *C*. Based on this definition, the similarity between two concepts can be formally defined as follows:

$$\text{sim}\,(c_1, c_2) = \max\,[IC(c)],\ c \in \text{Sup}(c_1, c_2),$$

where Sup(c_1, c_2) is a set of concepts that subsume both c_1 and c_2. Resnik only considered the information content of the least common subsumer (lcs), which has the shortest distance from the two concepts being compared. An *lcs* is a concept in a lexical taxonomy (e.g., WordNet).

Based on Resnik's information content approach, Lin (Lin, 1998) proposed a similarity theorem for measuring the distance between two concepts. The similarity between *A* and *B* is measured by the ratio between the amount of

information needed to state their commonality and that needed to fully describe them:

$$\text{sim(A,B)} = \frac{\log P(\text{common}(A,B))}{\log P(\text{description}(A,B))}$$

Hence, if we know the commonality of two concepts, their similarity indicates how much additional information is required to define the two concepts.

Couto et al. (2007) used the node-based approach to calculate semantic similarity between concepts in gene ontology (GO) through a graph-based similarity measure, called GraSM. They used all the information in the graph structure of the GO instead of considering it a hierarchical tree. Their measure of semantic similarity between GO terms demonstrated its greater effectiveness by obtaining a higher correlation using disjunctive common ancestors than using only the most informative common ancestor.

Edge-based approach

The edge-based approach is more natural and intuitive than the node-based approach for calculating semantic similarity in a semantic network. This approach estimates the distance between the nodes that correspond to the concepts being compared. Obviously, the shorter the distance between two nodes, the greater is their similarity.

Rada et al. (1989) as well as Rada and Bicknell (1989) adopted this approach for a hierarchical semantic network of terms used for indexing articles in the bibliographic retrieval system MEDLINE, which uses medical subject headings (MeSH). Their principal assumption for evaluating semantic similarity was that "the number of edges between terms in the MeSH hierarchy is a measure of conceptual distance between terms Rata et al., 1989, p. 18)." Based on this assumption, the distance between document *Doc* and query *Q* is defined as follows:

$$\text{DISTANCE}(Doc, Q) = \frac{1}{mn} \sum_{t_i \in Doc} \sum_{t_j \in Q} \text{d}(t_i, t_j),$$

where $\text{d}(t_i, t_j)$ is the minimal number of edges in a path from t_i to t_j within a hierarchy.

Lee et al. (1993) demonstrated an information retrieval method based on conceptual distance in IS-A hierarchies. They developed a method called the knowledge-based extended Boolean model, called KB-EBM, which involves a primitive distance function. Given terms t_i and t_j in an IS-A hierarchy, the distance (which is often called the primitive distance function) between the terms is as follows:

Distance (t_i t_j) = minimal number of IS-A relationships between t_i and t_j

- Statistical Method

The statistical method can be a good alternative for measuring similarity between

concepts. This method is used to compute similarity by analyzing auxiliary information, such as a corpus or dictionary, in a statistical way. Applications of pointwise mutual information (PMI) or latent semantic analysis (LSA) can be taken as representative of this method.

PMI of data collected by information retrieval (PMI-IR) was proposed by Turney (2001) as an unsupervised measure of semantic similarity of words. PMI-IR is based on word co-occurrence by using counts accumulated over very large corpora (e.g., the Web). Given two words w_1 and w_2, their PMI-IR is measured as follows:

$$\text{PMI-IR}(w_1,w_2) = \log_2 \frac{P(w_1 \& w_2)}{P(w_1)*P(w_2)}$$

which indicates their degree of statistical dependence and can be used as a measure of their semantic similarity.

Another statistical measure of semantic similarity is the result of the latent semantic analysis (LSA) proposed by Landauer et al. (1998). LSA is a "method for extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of text (p. 259)." In LSA, term co-occurrences in a corpus are captured by means of a dimensionality reduction and singular value decomposition of the term-by-document matrix representing the corpus.

12.2.2. Non-semantic measure (String Comparison)

- N-gram comparison

In computational linguistics, an n-gram is a continuous sequence of n letters from a given sequence of text or speech. An n-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram"; size 3 is a "trigram". Using this n-gram concept, similarity measure can be defined. N-gram string comparison measure is define as:

$$\frac{2 \cdot |n-grams(X) \cap n-grams(Y)|}{|n-grams(X)| + |n-grams(Y)}$$

Where n-gram(X) is a multi-set of letter n-grams in X.

- Jaro-Winkler Metric

The Jaro–Winkler distance (Winkler, 1990) is a measure of similarity between two strings. It is a variant of the Jaro distance metric (Jaro 1989, 1995), a type of string edit distance. The Jaro metric is not basd on edit distance model.

Given strings $s=a_1a_2...a_K$, and $t = b_1b_2...b_L$, define a character a_i in s to be common with t there is a $b_j = a_i$ in t such that $i-H \leq j \leq i+H$,

Where $H = \frac{\min(|s|,|t|)}{2}$. Let $s'=a'_1a'_2...a'_K'$, be the characters in s which are common with t (in the same order they appear in s) and let $t' = b'_1b'_2...b'_L'$, be analogous; now define a transposition for s', t' to be a position i such that $a'_i \neq b'_i$. Let $T_{s',t'}$ be half the number of transpositions for s' and t'. The Jaro similarity metric for s and t is

$$\text{Jaro}(s,t) = \frac{1}{3} \cdot \left(\frac{|s'|}{|s|} + \frac{|t'|}{|t|} + \frac{|s'| - T_{s',t'}}{|s'|}\right)$$

A variation of this uses the length P of the longest common prefix of s and t. Let *P'* = *max(P,4)*, Jaro-Winkler can be defined as:

$$\text{Jaro} - \text{Winkler}(\text{s}, \text{t}) = Jaro(s,t) + \frac{P'}{10} \cdot (1 - Jaro(s,t))$$

The Jaro and JaroWinker metrics are optimal for comparing short strings such as names.

- Levenshtein Distance and Ratio

In information theory, the Levenshtein distance is a string metric for measuring the difference between two string sequences. Informally, the Levenshtein distance between two words is the minimum number of single-character edits (i.e. insertions, deletions or substitutions) required to change one word into the other. It is named after Vladimir Levenshtein, who considered this distance in 1965 (Levenstein 1966).

$$lev_{a,b}(i,j) = \begin{cases} \max(i,j) & if\ \min(i,j) = 0 \\ \min \begin{cases} lev_{a,b}(i-1,j) + 1 \\ lev_{a,b}(i,j-1) + 1 \\ lev_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & otherwise. \end{cases}$$

Table below shows the comparison of three non-semantic string similarity measures.

Table 10. Comparisons of Representative Non-semantic String Similarity Measures

Measure	Properties
N-gram comparison (tri-gram)	- segmentation-based matching - n: size of segmentation (2~3)
Jaro-Winkler Metric	- character-transposition measure - best suited for short strings
Levenshtein Distance and Ratio (Sequence Matching)	- From DNA sequence matching algorithm - Character-by-character comparison

CHAPTER13. Experimental Evaluation of Learning-based Matcher Combination Strategy

13.1. Experiment Setting

13.1.1. Features for Classification Models

To evaluate the learning based schema matching algorithm, we set up the schema matching scenario and implement the matching framework using programming language Python 2.7 (Van Rossum 2010). Python language has many kind of strong library to implement various machine learning algorithms. In this dissertation, most of classifiers are implemented using scikit-learn[17] python library (Pedregosa et al. 2011). In addition to this library, many kind of computational linguistic approach is necessary to realize the data matching. To cover tasks related to computational linguists, NLTK library[18] is used (Bird et al. 2009).

Features candidate for learning models are selected based on the degree of explained variance. Total eight similarity measures are used for base features from semantic and non-semantic matching measures (see the list below).

- Semantic matching measure: path similarity, Wu-palmer measure, Lin measure, Resnik measure (Auxiliary data: WordNet, Rodget's thesaurus, Brown Corpus).

[17] This package is available at http://scikit-learn.org/stable/index.html

[18] This package is available at http://www.nltk.org/

- Non-semantic matching measure (Syntactic measure): Levenshtein distance, Jaro similarity, Jaro-winkler similarity, N-gram distance.

Some of semantic network-based measures need auxiliary data such as thesaurus, ontology, etc. We utilize the external data including WordNet, Rodget's thesaurus, and Brown Corpus from NLTK library.

In most of schema matching scenario, semantic matching measures fail to find correspondence between schemas elements as most of the element name are not found in their external data. An element name with abbreviated form should be pre-processed before the matching sequence is performed. In this case, however, each abbreviated element name should be extended and reformed by human, it violates the philosophy of automatic systems development. Most of semi-automatic matching systems have tended to load the pre-defined abbreviation dictionary to improve their matching quality. However, these approach strongly depends on the quality of abbreviation dictionary. In this dissertation, we did not include any kind of human-task at all.

As a result, we select three similarity measures as features for learning-based classification model; n-gram comparison, Jaro-Winkler metric, and Levenstein edit distance measures. Each of selected measures has lower inter-correlation because each of them adopt different property of string and sum of explained variance by three features are over 0.8.

13.1.2. Test data for Matching

Four our evaluation we used five XDR schemas for purchase orders, CIDX, Excel,

Noris, Paragon and Apertum as other representative prototypes used for evaluation. These data have common domain, purchase order, which has various schema elements to represent the property of purchase information in E-commerce environment. The detail specification of test data can be found in table below.

Table 11. Specification of Test Schema

Name	Type	# of nodes
CIDX	XDR	27
Excel	XDR	32
Noris	XDR	46
Paragon	XDR	59
Apertum	XDR	74

In addition to these data, we use other dataset used for evaluation of other research and prototypes. Test schema for purchase order is used to compare the performance of proposed method with existing representative algorithms. To show the robustness and generalizability of proposed approach, we used additional data and they will be explained in following section, Experimental Result.

13.1.3. Classification Algorithms for Schema Matching

There are many kind of machine learning algorithms and selecting appropriate algorithm that fits on the learning situation is one of the most important step in

machine learning application. We already review some representative algorithms for classification and their properties. In this section, the comparison between algorithms for schema matching is presented and discuss which algorithms will be appropriate for given schema matching situation. Kotsiantis (2007) scrutinized the strengths and drawbacks of supervised machine learning algorithms.[19] The comparison of supervised learning is presented in table below which is modified from Kotsiantis's (2007) comparison table.

Table 12. Comparing learning algorithms (** stars represent the best and * star the worst performance)**

	Decision Tree	Naïve Bayes	kNN	SVM	Rule-learner
Accuracy in general	**	*	**	****	**
Speed of learning w.r.t. number of attribute and instance	***	****	****	*	**
Speed of classification	****	****	****	*	**
Tolerance to missing value	***	****	*	**	**
Tolerance to highly interdependent attribute	**	*	*	***	**
Dealing with	****	***	***	**	***

[19] In this dissertation, only supervised learning will be dealt with as a schema matching problem needs the learning process to predict the label of test data.

discrete/binary/continuous attribute		not continuous	not discrete	not discrete	not continuous
Model Parameter Handing	***	****	***	*	***

As we can find in the table, naïve bayes and rule-learner are not applicable to continuous attribute case. Considering that the normalized semantic similarity measures are used for features, these two algorithms are not appropriate for schema matching task. In terms of overall accuracy, support vector machine show better performance but its learning speed and parameter handling might be critical weaknesses. kNN algorithms shows moderate performance in most of criteria. We choose three learning algorithms; kNN, SVM and Decision Tree (CART). To implement classification model, some parameters should be tuned to make prediction for test data. Table below show the parameters for each classification algorithm.

Table 13. Parameters for selected machine learning-based classifier

Common Parameter for Machine Learning		
Size of Training Set	~5%	

Major Parameter for Predict Model		
Algorithm	Parameter Type	Parameter Setting
K-nearest Neighbor (KNN)	- K - Weight	- 1~5 - Uniform/Distance
Support Vector Machine	- Kernel Type	- Linear - Gaussian (RBF)
Decision Tree (CART)	- Minimum sample leaf - Information Gain	- 2 - Gini impurity

13.2. Experiment Result of schema matching – CIDX and Excel purchase order schema

13.2.1. Result of matching based on kNN

First, we conduct schema matching as a classification with kNN algorithms. CIDX purchase order schema and Excel purchase order schema are used as source and target schema. In kNN, k is one of the most parameters and thumbs up rule is to select k as the square root of the number of features in uniform weight case. Total matching dimension is 27×32 and we take matching and un-matching pairs of schema element as sample training data. Training data is randomly selected from

matched and un-matched pairs and the evaluation[20] is repeated ten times. All the values are the average of ten training and test result.

- Uniform-weighted kNN

First, we take two matched and unmatched pairs as training data. As increase the number of training data, the evaluation criteria is examined. Details of experiment result is summarized in table below.

Table 14. The result of kNN with uniform weighting scheme (training size 2~5)

k	1	2	3
precision	0.3293379	0.4404863	0.413732
recall	0.7469481	0.6481061	0.7065079
f-measure	0.4305547	0.5194772	0.5178629

Sample size: 2 pairs of matched and unmatched schema elements set.

k	1	2	3	4	5
precision	0.3358095	0.5897243	0.425907	0.476343	0.3050838
recall	0.7897908	0.6394661	0.6991558	0.6594264	0.6838925
f-measure	0.4615186	0.6106573	0.52545	0.5506764	0.400974

Sample size: 3 pairs of matched and unmatched schema elements set.

[20] The evaluation methods of schema matching are adopted from Do et al. 2003.

k	1	2	3	4	5
precision	0.342791	0.5612904	0.4479364	0.3938944	0.4409692
recall	0.8058478	0.7347511	0.6972872	0.746075	0.7180628
f-measure	0.4801553	0.6345765	0.5385469	0.5117465	0.5419722

Sample size: 4 pairs of matched and unmatched schema elements set.

k	1	2	3	4	5
precision	0.3792674	0.5941673	0.3956395	0.4829418	0.3809936
recall	0.7837843	0.7037157	0.8368579	0.6947294	0.8081602
f-measure	0.4931309	0.6422512	0.5254973	0.5641508	0.516274

Sample size: 5 pairs of matched and unmatched schema elements set.

As the sample size increase, the overall evaluation measure is improved. In addition, we conclude that k is the critical factor affecting the performance of schema matching. Figure below shows the value of f-measure with respect to k and sample size.

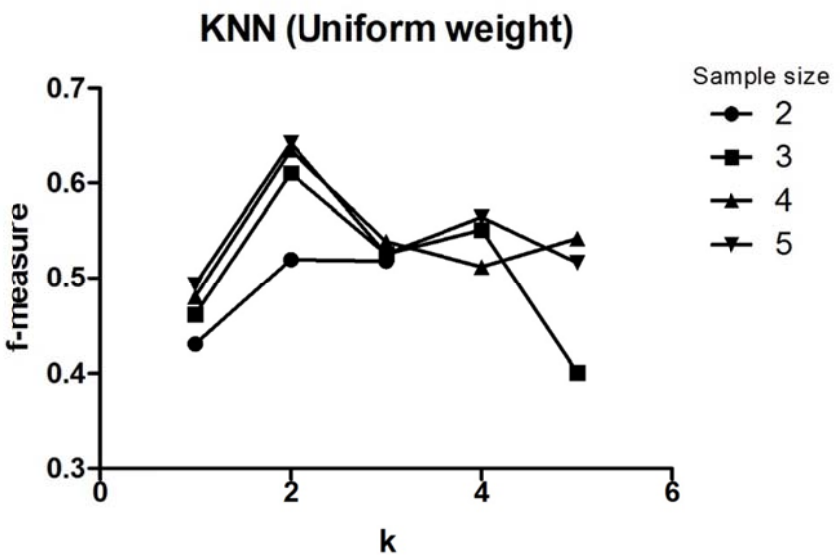


Figure 23. F-measure of kNN – uniform weighting scheme

In all sample size scenario, f-measure shows the highest value when k = 2. This result correspond to the thumbs up rule of determining k value in kNn; optimal k value is the square root of the features' number. In our case, we take three features and optimal k according to the thumbs up rule is approximately 2 ($\sqrt{3} \approx 1.73$).

- Distance-weighted kNN

For kNN classification, it can be useful to weight the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. A common weighting scheme consists in giving each neighbor a weight of 1/d, where d is the distance to the neighbor.The result based on this weighting schema is shown in table below.

Table 15. The result of kNN with distance-based weighting scheme (training size 2~5)

k	1	2	3
precision	0.3110721	0.3968701	0.4281441
recall	0.7360065	0.7317027	0.7548449
f-measure	0.4328101	0.5120707	0.5442223

Sample size: 2 pairs of matched and unmatched schema elements set.

k	1	2	3	4	5
precision	0.4214156	0.3858844	0.3978154	0.4777269	0.440949
recall	0.7330123	0.8031421	0.7492496	0.792684	0.7152994
f-measure	0.5314169	0.5173123	0.5154735	0.5947629	0.5406731

Sample size: 3 pairs of matched and unmatched schema elements set.

k	1	2	3	4	5	6
precision	0.3550972	0.3564426	0.3819648	0.5157942	0.5717456	0.447079
recall	0.7785642	0.8834668	0.7846681	0.7326082	0.6821537	0.785386
f-measure	0.4761777	0.506185	0.507714	0.6007633	0.6213285	0.5692857

Sample size: 4 pairs of matched and unmatched schema elements set.

k	1	2	3	4	5	6
precision	0.3913702	0.4502505	0.4065334	0.4381779	0.5275084	0.4894695
recall	0.7396501	0.8447258	0.810772	0.8395346	0.7588348	0.7734812
f-measure	0.5088621	0.5674944	0.5380513	0.5737644	0.6217626	0.5976742

Sample size: 5 pairs of matched and unmatched schema elements set.

In distance-based weight scheme, optimal k value is larger than uniform weight case[21]. Though f-measure shows first peak value in k=2, higher peak is detected in k = 5. The tables are summarized in figure below. In distance-based weighting scheme, k tends to have larger value than uniform weight scheme. k=4~5 has their peak value of f-measure.

[21] In distance-based weight scheme, generally, optimal k tends to be larger than uniform weight case.

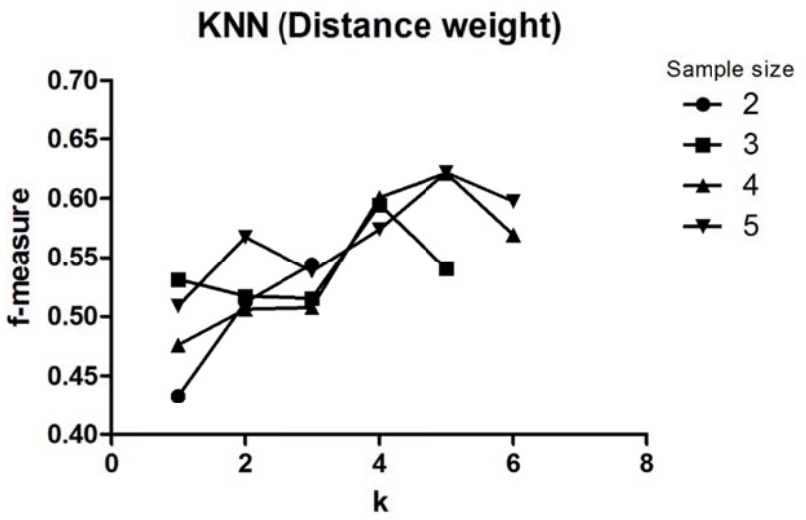


Figure 24. F-measure of kNN – distance-based weighting scheme

13.2.2. Result of matching based on SVM

First, we perform schema matching using SVM with linear kernel function, which is one of most commonly used kernel used by SVM approach. The result of schema matching with SVM is shown in table below.

Table 16. The Result of schema matching based on SVM with linear kernel function.

Sample Size	2	3	4	5	6	7
precision	0.164181	0.41703	0.521481	0.534855	0.534594	0.519313
recall	0.556483	0.691407	0.636364	0.636364	0.636364	0.658254
f-measure	0.2479	0.517438	0.57319	0.581159	0.581051	0.580473

As sample size increase, the overall performance of schema matching increase. The f-measure is saturated when the sample size is larger than 4. The experimental result is summarized in figure below.

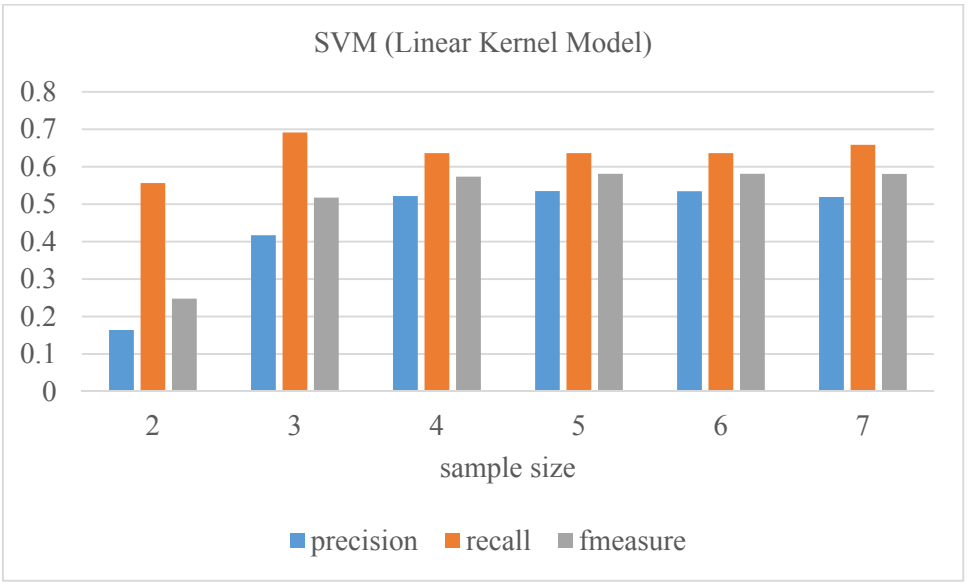


Figure 25. The result of SVM-based schema matching with linear kernel function

SVM with Gaussian rbf kernel also shows moderate matching result. It shows similar performance as SVM with a linear kernel. Result is summarized in table below.

Table 17. The Result of schema matching based on SVM with Gaussian rbf kernel function.

Sample Size	2	3	4	5	6	7
precision	0.32023	0.51504	0.520636	0.508301	0.523729	0.539468
recall	0.675263	0.636364	0.636364	0.636364	0.636364	0.636364
f-measure	0.421941	0.569273	0.572664	0.565157	0.574518	0.583859

Gaussian kernel model shows better performance in small sample size case. The 귤 kernel is one of the most popular kernel function.

$$f(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \exp(-\gamma ||\boldsymbol{x}_i - \boldsymbol{x}||^2) + b$$

Only with three samples of matched and unmatched sample, it approaches its best performance (see figure below).

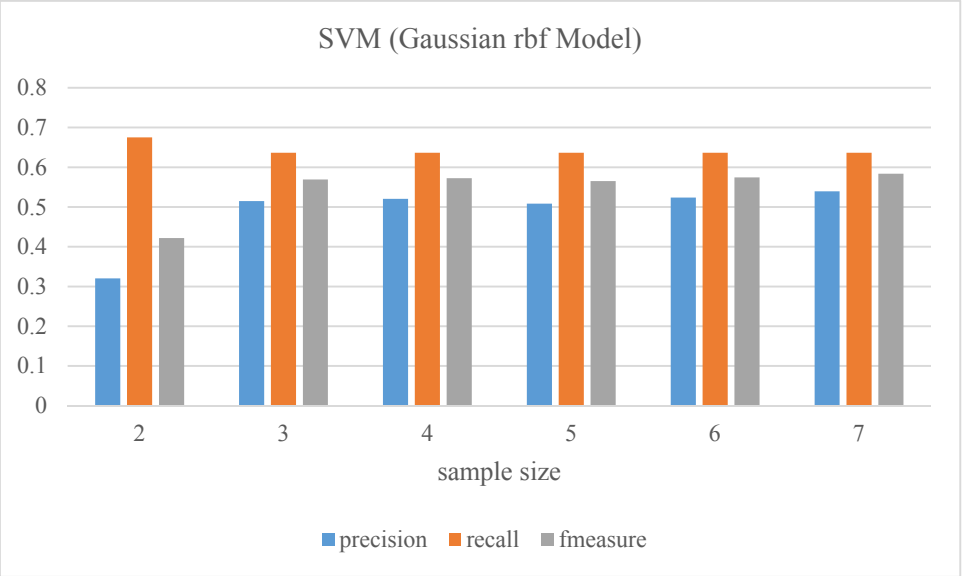


Figure 26. The result of SVM-based schema matching with rbf kernel function

13.2.3. Result of matching based on Decision Tree

There are many kind of decision tree algorithms. Each algorithm has its own fitness to classification situation and we should select the appropriate decision tree algorithms for schema matching research. Most representative decision tree algorithms are summarized in table below and we select the suitable algorithm based on this.

Table 18. Comparison of Representative Decision Tree Algorithms[22]

Method	CHAID	CART	C4.5
Splitting Criterion	Chi-square statistics	Gini Index	Information Gain Ratio
Merging process	Optimal grouping test for similarity	Binary Grouping	No merging
Recommendation Situation	Exploratory Phase Handling very large dataset	Classification with reliability No complicated settings	Small dataset Not sensitive to the setting
Not Recommended when	Classification performance matters	Small dataset	Three size increases with the dataset size

Based on the comparison table and the analysis of each decision tree algorithm in chapter 11, we select CART for schema matching task. In CART algorithm, Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it were randomly labeled according to the distribution of labels in the subset. Gini impurity can be computed by summing the probability of each item being chosen times the probability of a mistake in categorizing that item.

[22] This table is established based on the contents of Breiman et al. 1984 and http://en.wikipedia.org/wiki/Decision_tree

It reaches its minimum (zero) when all cases in the node fall into a single target category. To compute Gini impurity for a set of items, suppose i takes on values in {1, 2, ..., m}, and let f_i be the fraction of items labeled with value i in the set.

$$I_G(f) = \sum_{i=1}^{m} f_i(1 - f_i) = \sum_{i=1}^{m} (f_i - f_i^2) = \sum_{i=1}^{m} f_i - \sum_{i=1}^{m} f_i^2 = 1 - \sum_{i=1}^{m} f_i^2$$

The result of schema matching task with CART algorithm is shown in table below.

Table 19. The result of schema matching with CART decision tree algorithm

Sample size	2	3	4	5	6
precision	0.329523	0.280473	0.358585	0.364158	0.312764
recall	0.617594	0.816854	0.824351	0.919556	0.884412
f-measure	0.409497	0.416766	0.497769	0.519612	0.460797

As depicted in figure below, the result of CART is not good enough to be selected for most accurate learning algorithm. In most cases, the f-measure is under 0.5.

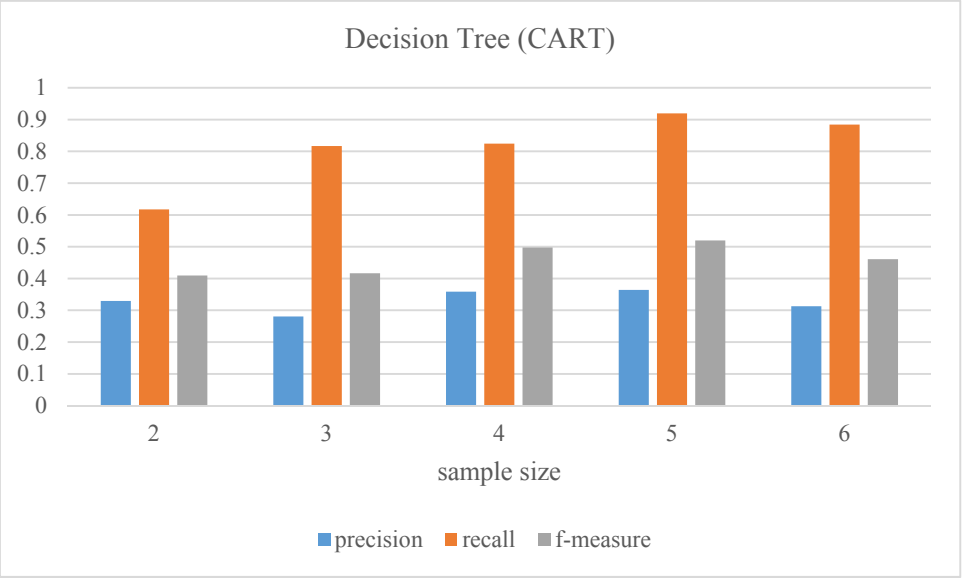


Figure 27. The result of schema matching with CART decision tree algorithm

In next section, we examine additional schema matching scenario to evaluate the applicability of proposed approach. More sample schemas of purchase order are used for evaluation and cross training-based schema matching is also performed to validate the general applicability of proposed approach.

13.2.4. Additional Experiment using various sample schemas

In this section, we evaluate the proposed approach using the purchase order schemas which are specified in section 13.1.2. Matching between Noris and Apertum, and Excel and Paragon purchase order schemas are performed.

- Matching between Noris and Apertum purchase order schema

The matching result of Noris and Apertum schema is presented in table below. The result of kNN and SVM-based matching shows higher performance than decision

tree-based matching. F-measure in most parameter settings has value about 0.8.

Table 20. Matching result – Noris and Apertum purchase order schema

kNN algorithms with uniform weighting scheme

training size: 2					
k	1	2	3		
precision	0.1857998	0.7312982	0.7297794		
recall	0.9052954	0.7743122	0.857134		
f-measure	0.306809	0.7510486	0.7881949		
training size: 3					
k	1	2	3	4	
precision	0.5078518	0.6944944	0.6932934	0.7272727	
recall	0.8916667	0.739224	0.870097	0.8137213	
f-measure	0.6350278	0.7138001	0.7711896	0.7668211	
training size: 4					
k	1	2	3	4	5
precision	0.2630678	0.6275179	0.729502	0.6775784	0.6964164
recall	0.8835538	0.7468827	0.7877601	0.7580071	0.8767284
f-measure	0.3935309	0.672593	0.7570643	0.713245	0.7760244

kNN algorithms with distance-based weighting scheme

training size2					
k	1	2	3		
precision	0.3106053	0.52737	0.6489827		
recall	0.828157	0.7382716	0.8418122		
f-measure	0.4482473	0.5893666	0.727457		
training size 3					
k	1	2	3	4	
precision	0.421176	0.4493383	0.4966667	0.635835	
recall	0.8619004	0.8423765	0.9172619	0.9108289	
f-measure	0.5535746	0.5457745	0.6303199	0.7472425	
training size4					
k	1	2	3	4	5
precision	0.265058	0.5399462	0.5008892	0.6617955	0.6221111
recall	0.8828527	0.8344136	0.9058642	0.8639859	0.9176323
f-measure	0.405543	0.6312971	0.6369849	0.7490929	0.7366581

SVM algorithms with RBF Gaussian kernel scheme

Sample size	2	3	4	5	6
precision	0.5457856	0.7272727	0.7307364	0.7091873	0.7297794
recall	0.8624559	0.8888889	0.8460229	0.8142681	0.857134
f-measure	0.6461656	0.8	0.7838706	0.7578765	0.7881949

SVM algorithms with linear kernel scheme

Sample size	2	3	4	5	6
precision	0.6777037	0.7199042	0.7202097	0.654464	0.7096138
recall	0.8825309	0.857134	0.8361199	0.8888889	0.809515
f-measure	0.7618621	0.7824837	0.77363	0.7529414	0.7561343

Decision Tree

Sample size	2	3	4	5	6
precision	0.2795227	0.5532215	0.3732528	0.3746102	0.3536245
recall	0.8811111	0.8227646	0.89847	0.9233422	0.9460229
f-measure	0.4209203	0.6483031	0.503959	0.5067073	0.4866516

- Matching between Excel and Paragon purchase order schema

The matching result of Excel and Paragon schema is presented in table below. The result of kNN and SVM-based matching shows higher performance than decision tree-based matching. F-measure in most parameter settings has value about 0.6.

Table 21. Matching result – Excel and Paragon purchase order schema

kNN algorithms with uniform weighting scheme

training size: 2					
k	1	2	3		
precision	0.3366938	0.390994	0.6177373		
recall	0.8416534	0.7154012	0.5555556		
f-measure	0.4785952	0.4919214	0.5849675		
training size: 3					
k	1	2	3	4	
precision	0.3990077	0.5880104	0.5840796	0.6056242	
recall	0.8865256	0.6586023	0.5790697	0.5780996	
f-measure	0.5401849	0.6198101	0.577881	0.5912478	
training size: 4					
k	1	2	3	4	5
precision	0.3970918	0.5865822	0.4544104	0.5421414	0.5046022
recall	0.9157496	0.7273942	0.7755423	0.6044356	0.659515
f-measure	0.5377443	0.6405477	0.5693481	0.5680789	0.5670752

kNN algorithms with distance-based weighting scheme

training size2					
k	1	2	3		
precision	0.2602081	0.3956572	0.3808386		
recall	0.8724471	0.8257981	0.7749956		

f-measure	0.395546	0.5151093	0.4964542		
training size 3					
k	1	2	3	4	
precision	0.3999108	0.3557998	0.6477056	0.5008798	
recall	0.8925	0.9067416	0.7483995	0.7170062	
f-measure	0.5461664	0.5075122	0.6872561	0.58513	
training size4					
k	1	2	3	4	5
precision	0.3241237	0.3646668	0.4379063	0.5747237	0.4319068
recall	0.9356746	0.9118695	0.8225132	0.7559965	0.9551146
f-measure	0.4793013	0.5160148	0.5668651	0.6432291	0.5919058

SVM algorithms with RBF Gaussian kernel scheme

Sample size	2	3	4	5	6
precision	0.4302605	0.5714303	0.6020136	0.6150949	0.625
recall	0.6880335	0.5792989	0.5772002	0.5602469	0.5555556
f-measure	0.5169915	0.572625	0.5878025	0.5860347	0.5882353

SVM algorithms with linear kernel scheme

Sample size	2	3	4	5	6
precision	0.4455017	0.559456	0.6783365	0.607415	0.6342287

recall	0.7008201	0.6511905	0.5555556	0.5625926	0.5743474
f-measure	0.5386257	0.596845	0.6098952	0.5831116	0.6027498

Decision Tree

Sample size	2	3	4	5	6
precision	0.282293	0.442778	0.3709205	0.4432737	0.4272306
recall	0.7378086	0.7235141	0.8868034	0.8874912	0.9074647
f-measure	0.3984665	0.5144991	0.5003226	0.5882848	0.5693641

13.2.5. Validation of classification using cross domain training data

In this section, we examine two case of matching task for validation of general applicability of proposed approach and reusability of matching result by performing cross training. In first experiment, matching between Noris and Apertum schema again using the training data from matching between Excel and Paragon data; this is cross training test using same domain schema. In second experiment, Noris and Apertum schema again using the training data from matching between university course schema data; this is cross training test using different domain schema. By comparing same matching among cross training using same and different domain, the general applicability can be proven.

1st Cross training test using same domain schema

Training data: matching data between excel and paragon data

Test data: Noris and Apertum

Table 22. Cross training result using same domain schema

kNN algorithms with uniform weighting scheme

training size: 2					
k	1	2	3		
precision	0.3641412	0.6949583	0.6766336		
recall	0.8982848	0.8841975	0.8524427		
f-measure	0.4939442	0.7778631	0.7536835		
training size: 3					
k	1	2	3	4	
precision	0.4488458	0.614763	0.721212	0.7192258	
recall	0.8050088	0.7437213	0.888889	0.8598501	
f-measure	0.5613103	0.6694753	0.79619	0.7832225	
training size: 4					
k	1	2	3	4	5
precision	0.4160755	0.6225063	0.7035912	0.5856047	0.5561895
recall	0.8369356	0.8782451	0.7762434	0.8888889	0.89
f-measure	0.5473442	0.7275507	0.7348946	0.7045936	0.6765601

kNN algorithms with distance-based weighting scheme

training size2					
k	1	2	3		
precision	0.3888424	0.6651605	0.6325915		
recall	0.8598501	0.8410582	0.8888889		
f-measure	0.5230336	0.7425697	0.7375192		
training size 3					
k	1	2	3	4	
precision	0.3982407	0.3919317	0.4672407	0.5000771	
recall	0.8073545	0.8745414	0.8782451	0.857134	
f-measure	0.5293606	0.5408826	0.6033213	0.6268143	
training size4					
k	1	2	3	4	5
precision	0.4088922	0.4015249	0.4881796	0.6154821	0.4931441
recall	0.8888889	0.8745414	0.8912346	0.870097	0.8814198
f-measure	0.5586455	0.5429832	0.6271223	0.7201723	0.6294281

SVM algorithms with RBF Gaussian kernel scheme

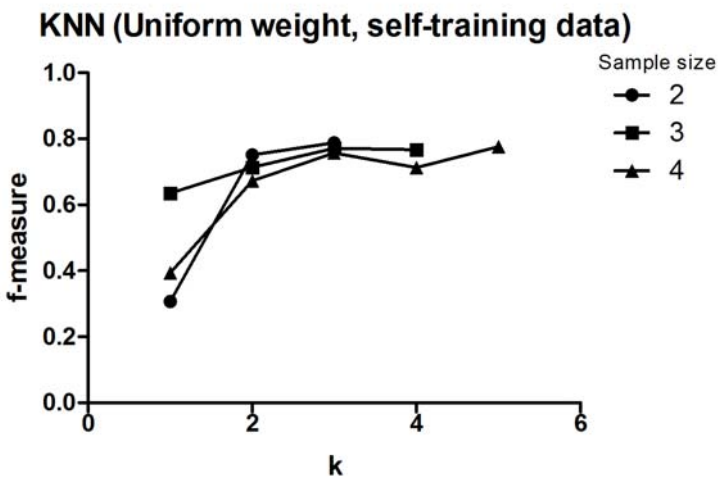
Sample size	2	3	4	5	6
precision	0.4075861	0.6747582	0.6921335	0.7255711	0.7030071
recall	0.8704189	0.8888889	0.8888889	0.8814198	0.8888889
f-measure	0.5255099	0.7657822	0.7780517	0.7959206	0.7849929

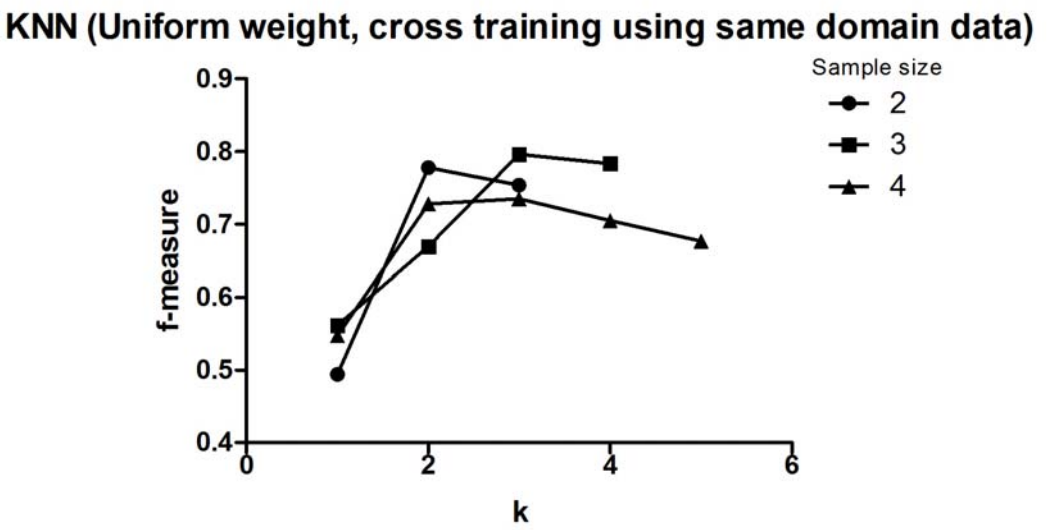
SVM algorithms with linear kernel scheme

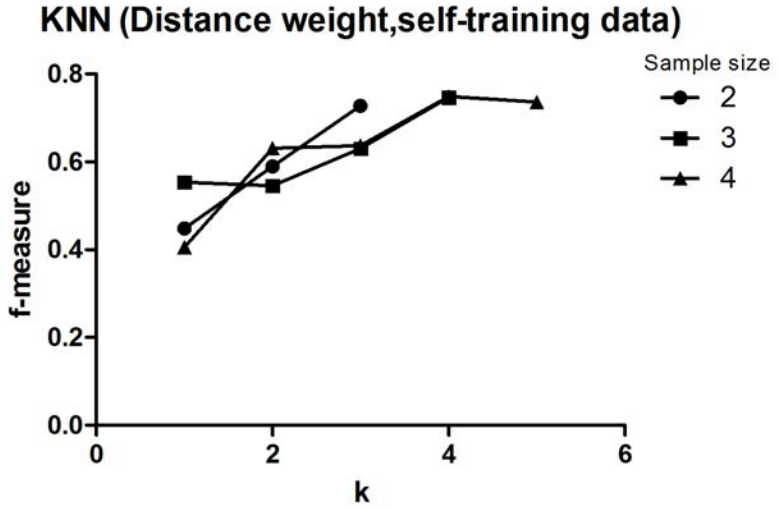
Sample size	2	3	4	5	6
precision	0.4932297	0.6815222	0.7081863	0.6766904	0.7212148
recall	0.8888889	0.8866667	0.8116402	0.825873	0.8383422
f-measure	0.630598	0.7685195	0.7559878	0.7430147	0.7752158

Decision Tree

Samle size	2	3	4	5	6
precision	0.313823	0.4760784	0.2548063	0.3503382	0.2262174
recall	0.8332628	0.8782451	0.8475529	0.8767284	0.8814198
f-measure	0.4473516	0.615863	0.3769863	0.4984244	0.3583832




KNN (Uniform weight, cross training using same domain data)
Sample size
2
3
4
f-measure
0.9
0.8
0.7
0.6
0.5
0.4
0
2
4
6
k


KNN (Distance weight,self-training data)
Sample size
2
3
4
f-measure
0.8
0.6
0.4
0.2
0.0
0
2
4
6
k

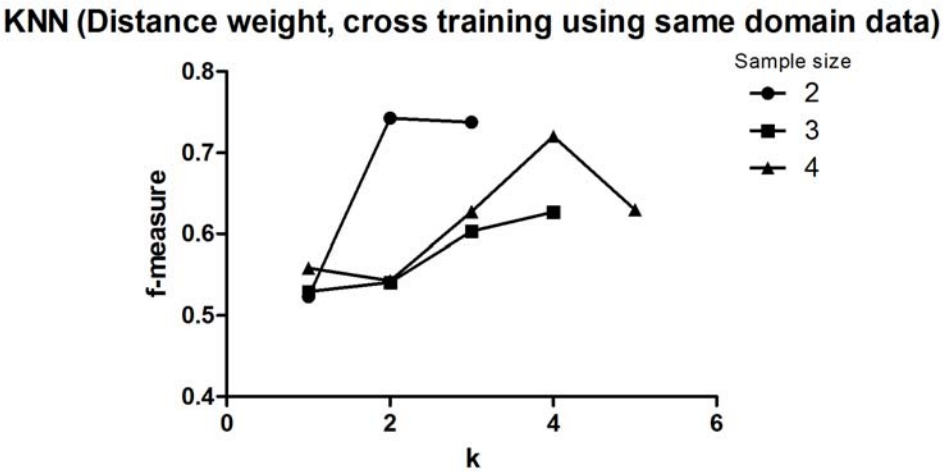
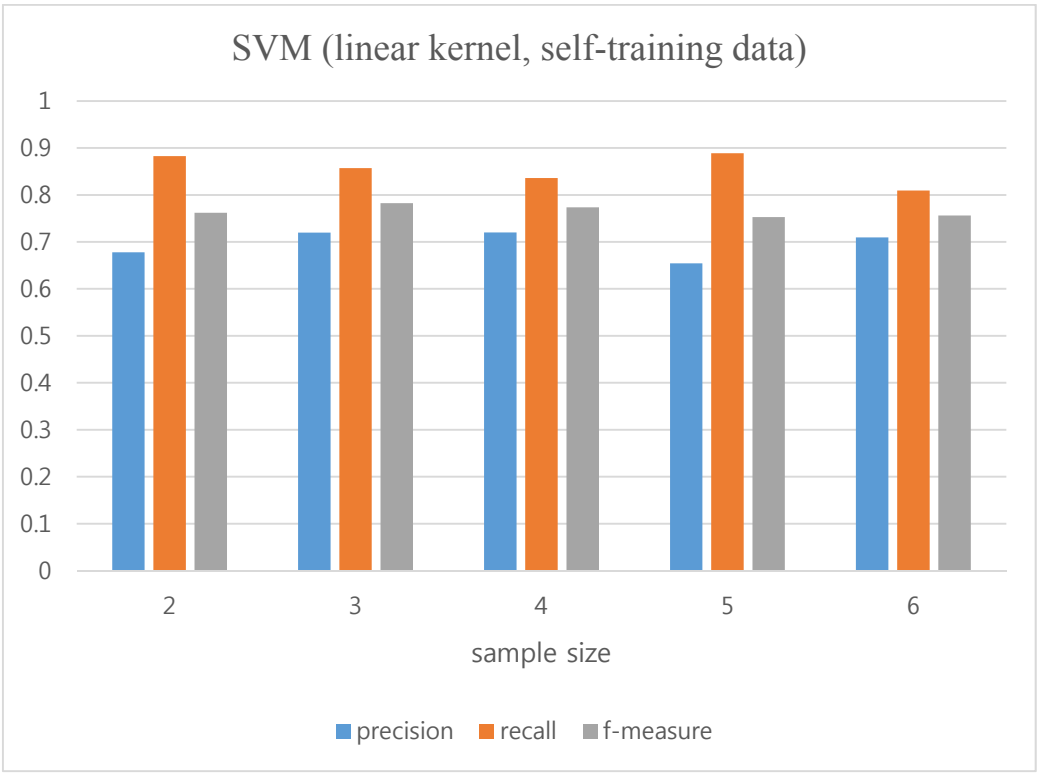
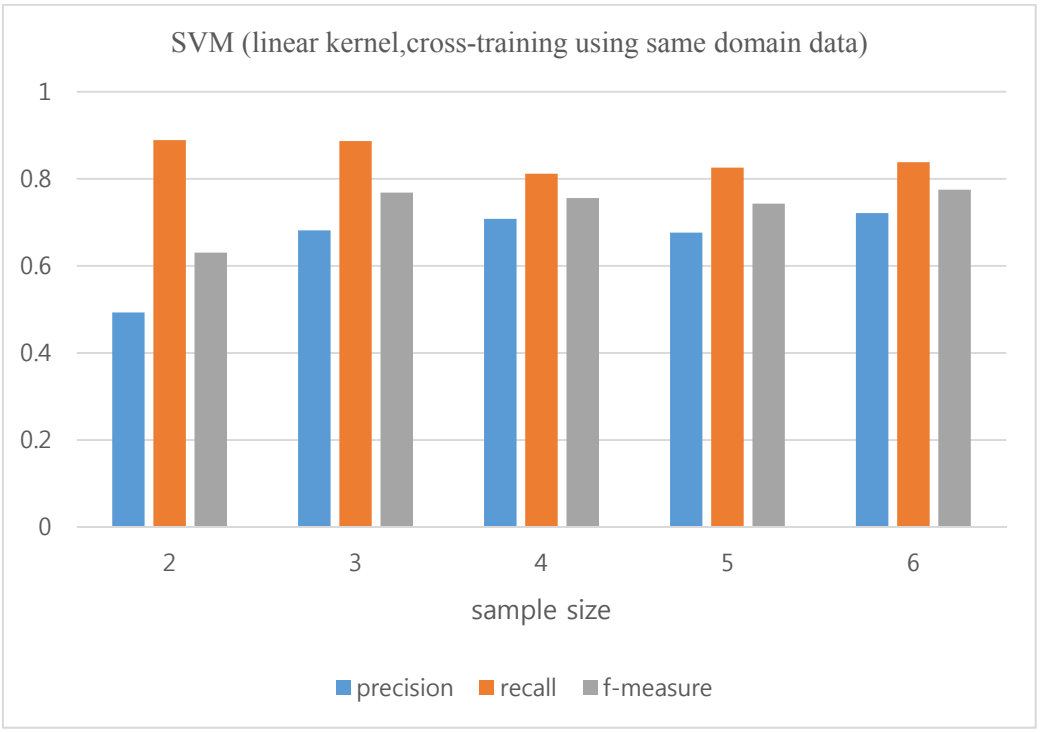
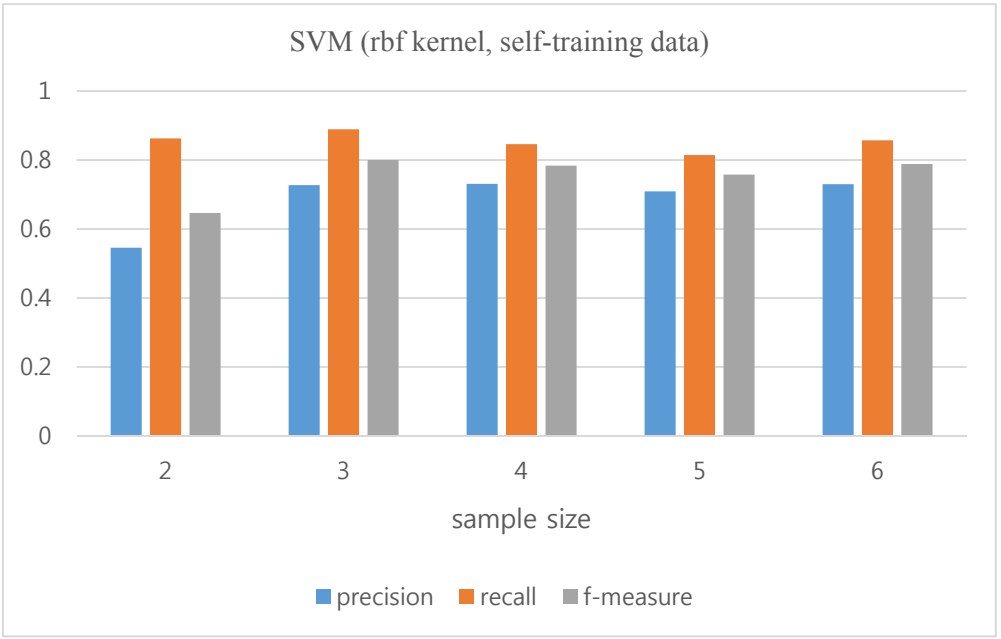


Figure 28. Comparison of Result: self-training vs. cross-training using same domain data (kNN classifier)

In the case of matching based on kNN, a cross-training result shows the same performance as self-training scenario. SVM also shows no difference between self-training and cross training. The details of comparison are shown in figure below.


SVM (linear kernel, self-training data)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
2
3
4
5
6
sample size
precision
recall
f-measure


SVM (linear kernel,cross-training using same domain data)
1
0.8
0.6
0.4
0.2
0
2
3
4
5
6
sample size
precision
recall
f-measure


SVM (rbf kernel, self-training data)
1
0.8
0.6
0.4
0.2
0
2
3
4
5
6
sample size
precision
recall
f-measure

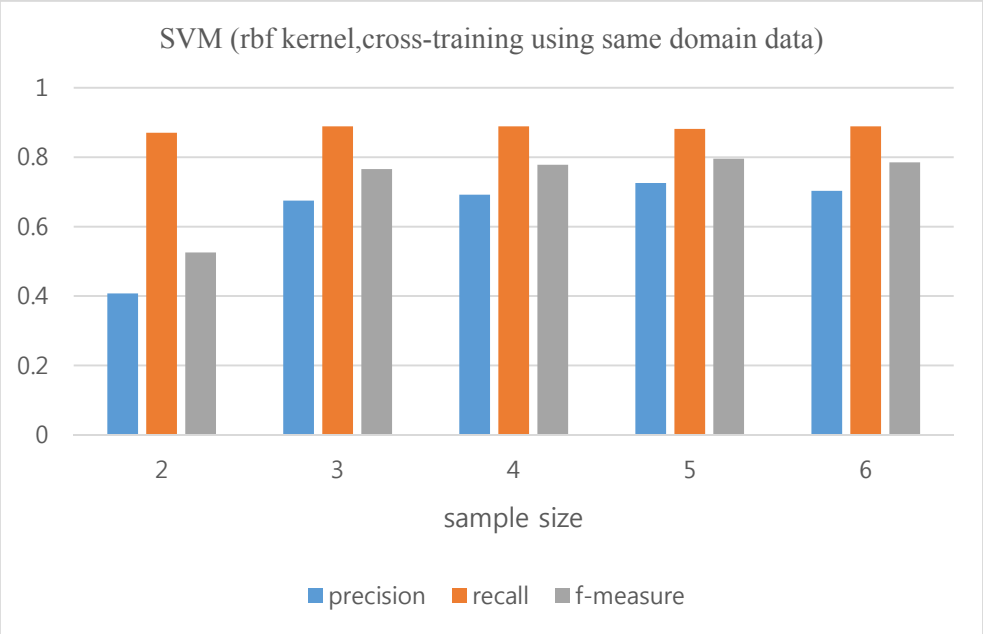


Figure 29. Comparison of Result: self-training vs. cross-training using same domain data (SVM classifier)

This experiment result implies that there needs no additional training process to cope with another schema matching problem from same domain. SVM shows the most robust result among three classification algorithms. Regardless of the kernel type and sample size, SVM's performance is better than other algorithms.

2nd Cross training test using different domain schema

In this part, we compare the matching result between self-training and cross training using other domain's training set. If there's no difference between two

cases, the proposed algorithm can cope with various matching situation without additional training data set. We perform a same schema matching task using the training data from different domain. A university course schema is used for this training data. Matching task is Noris and Apertum purchase order schema matching again. The result shown in table and figure below, cross training using different domain data also show very nice performance comparing self and cross training using same domain-based matching.

- Cross training using different domain schema

Training data: University course schemas

Test data: Noris and Apertum

Table 23. Cross training result using different domain schema

kNN algorithms with uniform weighting scheme

training size: 2					
k	1	2	3		
precision	0.536711	0.656057	0.66179		
recall	0.888889	0.813157	0.888889		
f-measure	0.668196	0.723397	0.758377		
training size: 3					
k	1	2	3	4	

precision	0.4671579	0.716196	0.6552538	0.593394	
recall	0.8645414	0.835317	0.8701235	0.888889	
f-measure	0.6063028	0.771038	0.7472496	0.710461	
training size: 4					
k	1	2	3	4	5
precision	0.4837717	0.6852538	0.653062	0.6292472	0.6698836
recall	0.8645414	0.8316755	0.888889	0.8888889	0.8238007
f-measure	0.6200357	0.7509845	0.752795	0.7347	0.737757

kNN algorithms with distance-based weighting scheme

training size2					
k	1	2	3		
precision	0.492392	0.5942564	0.6180584		
recall	0.8314991	0.8888889	0.8888889		
f-measure	0.6170279	0.7112816	0.7281156		
training size 3					
k	1	2	3	4	
precision	0.5429531	0.5237415	0.6423999	0.6001321	
recall	0.8094533	0.8888889	0.857134	0.8841975	
f-measure	0.6489629	0.6566812	0.7337303	0.7136363	
training size4					

k	1	2	3	4	5
precision	0.528983	0.5913208	0.6241401	0.6076837	0.6144182
recall	0.8841975	0.8888889	0.8888889	0.8888889	0.8814198
f-measure	0.6610621	0.7079842	0.7327551	0.7212625	0.7219626

SVM algorithms with RBF Gaussian kernel scheme

Sample size	2	3	4	5	6
precision	0.5613785	0.6335486	0.7087786	0.7087786	0.6867875
recall	0.8626279	0.857134	0.8888889	0.8888889	0.870097
f-measure	0.6792059	0.7276216	0.7885826	0.7885826	0.7673657

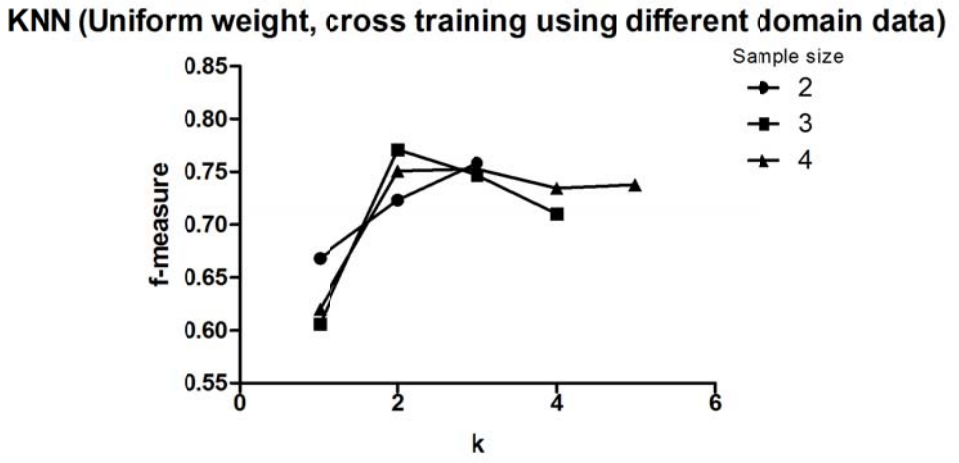
SVM algorithms with linear kernel scheme

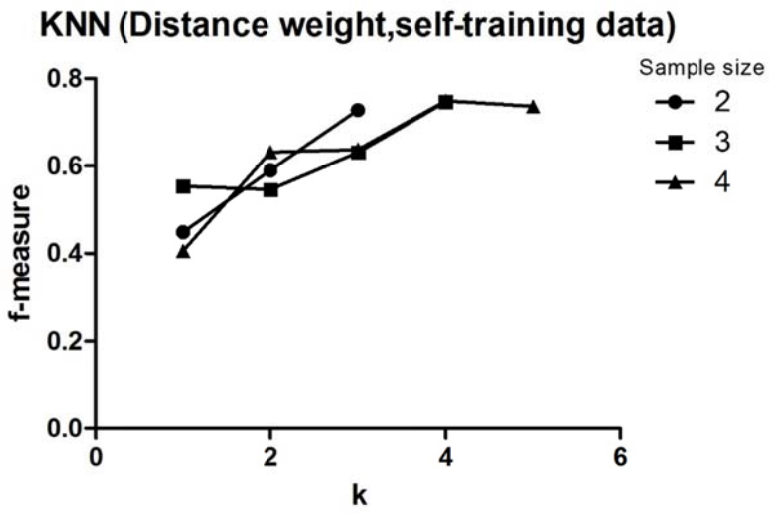
Sample size	2	3	4	5	6
precision	0.5840708	0.7159151	0.7037065	0.7111163	0.7033891
recall	0.8460229	0.8814198	0.8888889	0.8238007	0.857134
f-measure	0.6879048	0.7900673	0.7851975	0.7628346	0.7724963

Decision Tree

Sample size	2	3	4	5	6
precision	0.4440433	0.5092772	0.5252076	0.434741	0.4530056

recall	0.7957804	0.8814198	0.7571781	0.8645414	0.8888889
f-measure	0.5498026	0.6453449	0.6188473	0.5696214	0.5961697





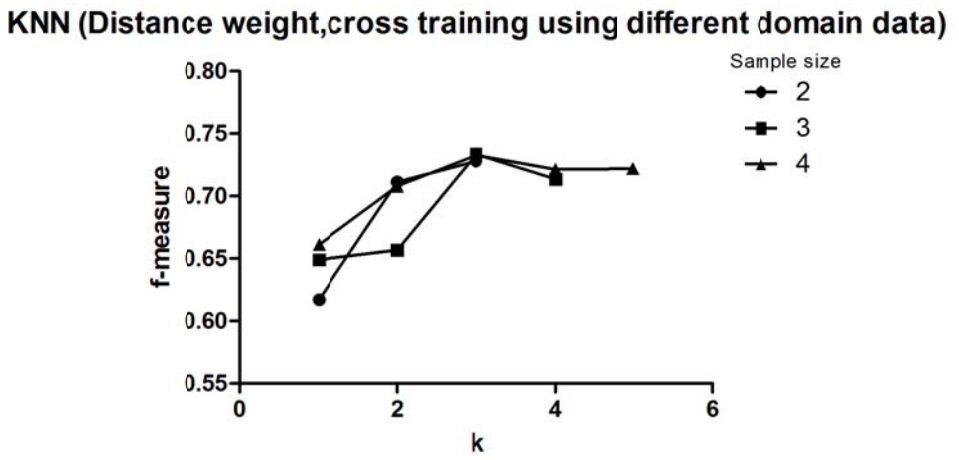
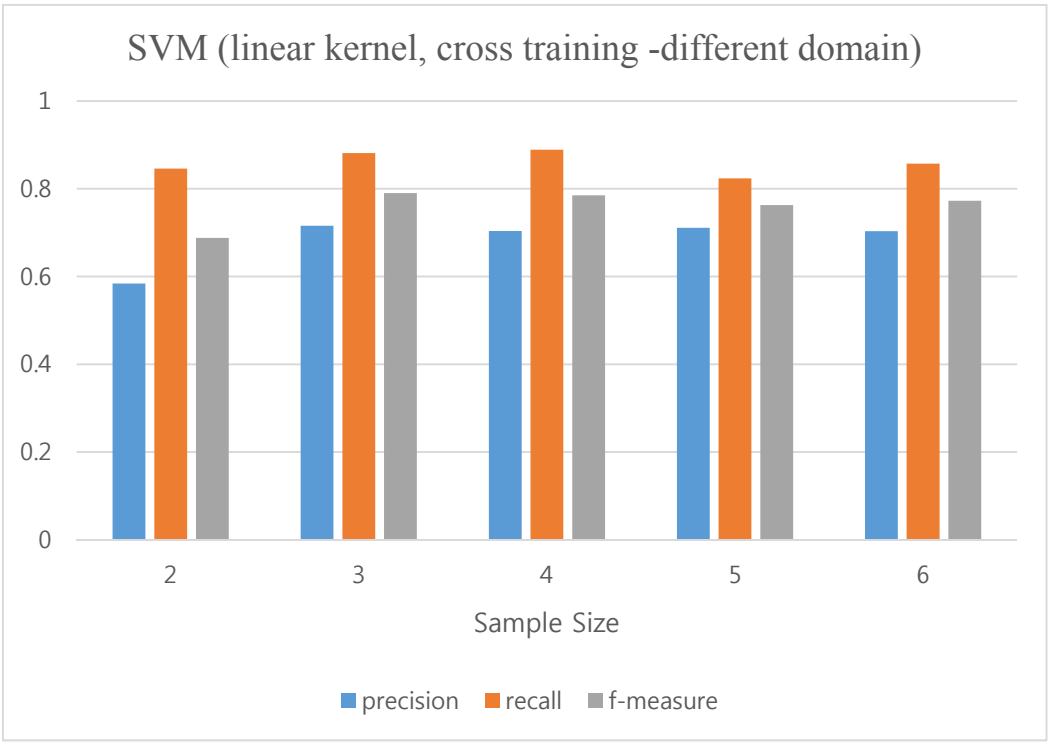
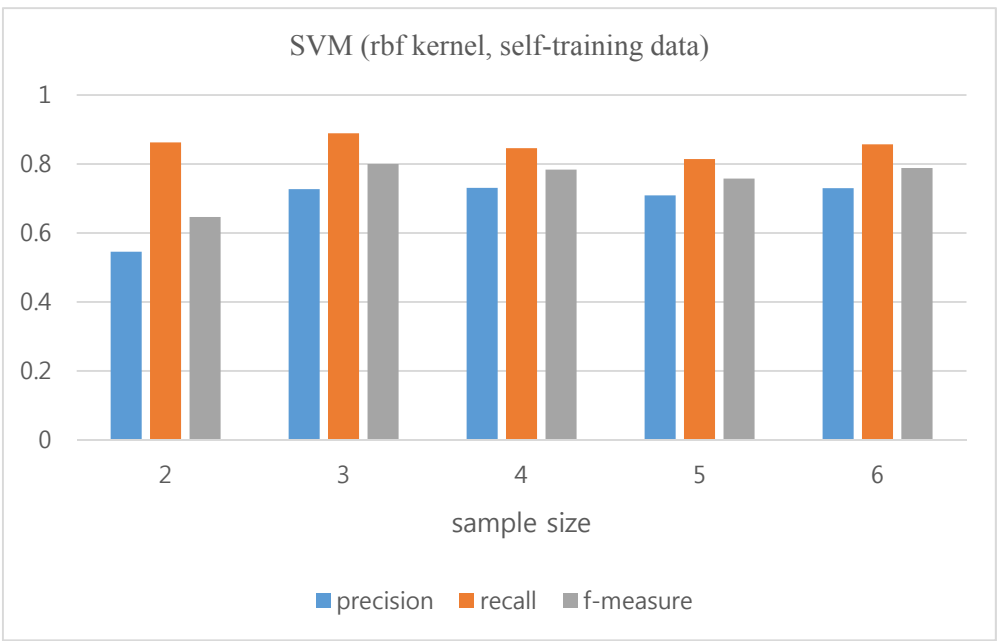


Figure 30. Comparison of Result: self-training vs. cross-training using different domain data (kNN classifier)

In the case of matching based on kNN, a result for cross-training using different domain training data shows the same performance and as self-training scenario.

SVM also shows no difference between self-training and cross training. The details of comparison are shown in figure below.


SVM (linear kernel, cross training -different domain)
1
0.8
0.6
0.4
0.2
0
2
3
4
5
6
Sample Size
precision
recall
f-measure


SVM (rbf kernel, self-training data)
1
0.8
0.6
0.4
0.2
0
2
3
4
5
6
sample size
precision
recall
f-measure

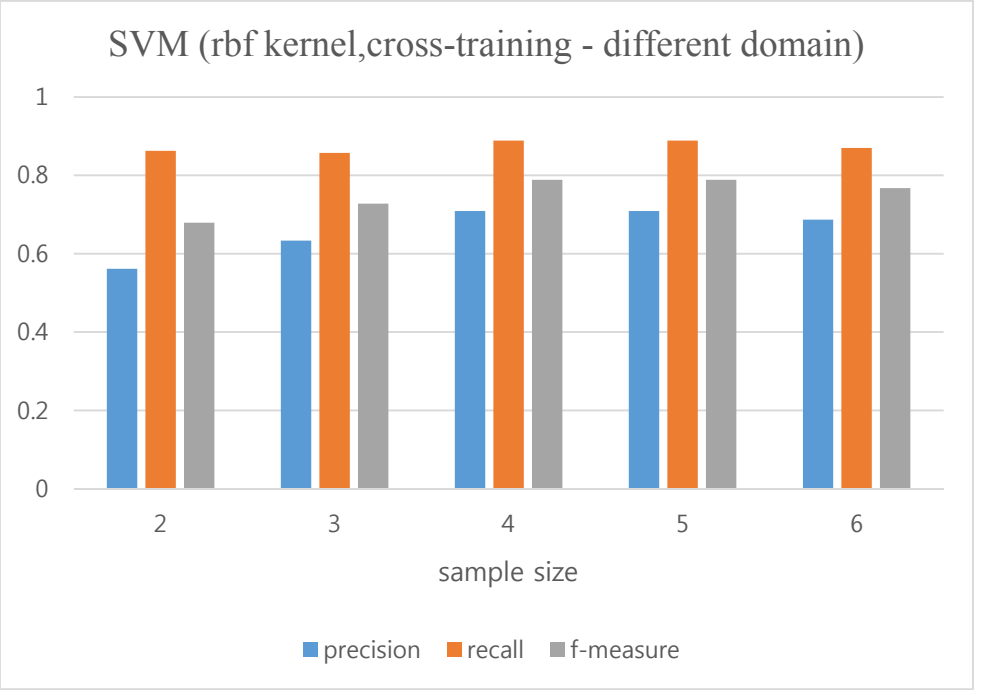


Figure 31. Comparison of Result: self-training vs. cross-training using different domain data (SVM classifier)

13.3. Discussions for mater combination strategy

13.3.1. Overall Performance of Matcher Combination Strategy

Overall matching performance of learning-based matcher combination show good performance in terms of evaluation metric such as f-measure. In most case, the f-measure has its range between 0.6~0.8, which is same as the performance of state of art architecture (semi-automatic) of schema matching such as COMA++.

Especially, in small schema matching (the size of matching element is less than 20), the matching performance metric shows very high value over 0.9. In this

dissertation, most of sample schema has less than 50 schema elements. In matching experiment using these sample schema, the proposed approach also show good performance comparing other schema matching approach. The summary of overall experiment in this dissertation is shown in table below.

Table 24. Experiment Summary

Domain	#of Schema (# of matching task)	#of element	Performance* (range of F-measure)
Purchase order	5 ($\binom{5}{2}$)	27~74	0.58~0.80
Auction	5 ($\binom{5}{2}$)	8~13	0.62~0.87
University Course	18 ($\binom{18}{2}$)	6~12	0.71~1.0
Bio-industry personal information	2 (1)	24	0.53

* In most case, SVM with RBF Gaussian kernel shows best performance.

13.3.2. Validity of Domain-cross training and Generalizability of Matcher Combination Strategy

If there exists the difference among training using own data, different data from same domain, and from other domain, the proposed approach implies that the classification model should be trained before every matching tasks. As we examine through additional experiment, however, most of matching scenario shows the robust result regardless of the way of training. This means that only the training using small quantity of matching result can be very useful for other matching regardless of their domain.

PART5. CLOSING

CHAPTER 14. Conclusion

In this dissertation, a cross similarity vector approach and learning-based matcher combination strategy are suggested. The contribution of this dissertation can be seen as two aspects; measure itself and combination strategy.

In terms of measure for schema matching itself, cross similarity vector approach improve the performance of structural measure comparing with existing other structural ones. By adopting the concepts of context in database research, a context-based similarity between schema elements is suggested for finding the semantic structural correspondence. Existing structural matcher only considers the morphological identity and the same elements who have different morphologies are considered as non-matched pair. In contrast, proposed approach calculates the affinity between the contexts of schema elements. The context of each elements is considered so that the measure can reflect the semantic structure of schema element. In addition, cross similarity vector approach enables to make another measure corresponding with specific string-based similarity measure. As cross similarity vector approach uses the similarity function between the textual information of schema elements, the approach can generate the structural measure using given string-based measure. Considering that the number of matchers is closely related to the performance of matching in multiple matcher approach, the approach can improve the matching performance with given string-based measures.

Hundreds of matcher has been proposed since the schema matching research has been dealt with. It has not been so long time since many researchers realize that there is no matcher which shows the best performance in every case of matching. Most of proposed single matchers are implemented for a specific matching situation. To cope with the general case of schema matching tasks, the strategy for combining existing various matchers is necessary. Proposed learning-based matcher combination strategy can be the keystone of multiple matcher approach. Learning-based matcher combination strategy realized the fully automatic schema matching by selecting existing measures dynamically. Without any human intervention, this approach show as great performance as semi-automatic shows. By transforming the multi-matchers based schema matching problem into binary classification problem, the complexity of combination problem is dramatically decreased. A classification approach has many strengths compared with matcher combination problem. The complexity of matcher combination increases in proportional to the number of matchers.

The future research will focus on the combination strategy for multiple matcher-based data matching. The learning-based matcher combination strategy can be applied to other situations dealt with data matching problem such as data field matching and entity resolution. Most data matching research adopt various matching metric for data comparison. As schema matching research uses many kind of matching metric, data matching needs various matcher according to the matching situation. In this regards, the general framework for data matching can be

implemented based on the proposed learning-based matcher combination strategy.

Therefore, the main focus of future research will lies in the general framework for data matching using the learning-based matcher selection strategy. Then this approach can be applied to various specific data matching problem. There are still many kind of data which should be matched for specific purposes. With the big wave of big data, I believe, proposed dissertation give the valuable intuition for data matching research.

APPENDIX A. PYTHON LIBRARIES AND DEPENDENCIES AMONG THEM.

The scikit-learn libraryh has the following dependencies (Pedregosa et al. 2011):

- numpy: this is a python module which has powerful tools for the creation and manipulation of arrays. It is the foundation of most scientific computing packages in python
- scipy: this is a python module which builds on numpy and provides fast implementations of many basic scientific algorithms.
- matplotlib: this is a powerful package for generating plots, figures, and diagrams. Our main form of visual interaction with data and results depends on matplotlib.

References

Algergawy, A., Schallehn, E., & Saake, G. (2009). Improving XML schema matching performance using Prüfer sequences. *Data & Knowledge Engineering*, 68(8), 728-747.

Algergawy, A., Nayak, R., & Saake, G. (2010). Element similarity measures in XML schema matching. *Information Sciences*, 180(24), 4975-4998.

Algergawy, A., Massmann, S., & Rahm, E. (2011, January). A clustering-based approach for large-scale ontology matching. In Advances in Databases and Information Systems, Springer Berlin Heidelberg. 415-428

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician* 46(3), 175–185. doi: 10.1080/00031305. 1992.10475879.

Atzeni, P. (2007, January). Schema and data translation: A personal perspective. In Advances in Databases and Information Systems. Springer Berlin Heidelberg. 14-27.

Aumueller, D., Do, H. H., Massmann, S., & Rahm, E. (2005). Schema and ontology matching with COMA++. *in Proceedings of the 2005 ACM SIGMOD International Conference on Management of data*, 906-908.

Batini, C., Lenzerini, M., Navathe, S.B. (1986). A comparitive analysis of methodologies for database schema integration. *ACM Computing Surveys,*

18(4), 323–364

Bellahsene, Z., Bonifat, A., and Rahm, E. (Eds.) (2011) Schema Matching and Mapping. Springe, Berlin, Heidelberg.

Bergamaschi, S., Castano, S., Vincini, M., & Beneventano, D. (2001). Semantic integration of heterogeneous information sources. Data & Knowledge Engineering, 36(3), 215-249.

Bernstein, Philip A., and Rahm, E. (2000). Data Warehouse Scenarios for Model Management. In *Proceedings of Conceptual Modeling-ER 2000: 19th International Conference on Conceptual Modeling*, Salt Lake City, Utah, USA, October 9-12, 2000, Vol. 1920. Springer.

Bernstein, P. A., Halevy, A., and R. A. Pottinger. (2000) A Vision for Management of Complex Models. *ACM SIGMOD Record*, 29(4), 55-63.

Bernstein, P. A. (2001). Generic model management a database infrastructure for schema manipulation. Cooperative Information Systems, Springer, 1-6.

Bernstein, P. A., and S. Melnik. (2004). Meta data management. In *Proceedings of the IEEE CS International Conference on Data Engineering*. IEEE Computer Society.

Bernstein, P. A., Madhavan, J. and Rahm, E. (2011). Generic schema matching, ten years later. In *Proceedings of the VLDB Endowment 4.11,* 695-701.

Bertino, E., Guerrini, G., & Mesiti, M. (2004). A matching algorithm for measuring

the structural similarity between an XML document and a DTD and its applications. *Information Systems*, 29(1), 23-46.

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

Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O'Reilly Media Inc.

Bohannon, P., Elnahrawy, E., Fan, W., & Flaster, M. (2006, September). Putting context into schema matching. In Proceedings of the 32nd international conference on Very large data bases,VLDB Endowment. 307-318

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and regression trees. CRC press.

Casanova, M. A., Breitman, K. K., Brauner, D. F., and Marins, A. L. (2007). Database conceptual schema matching. *Computer*, 40(10), 102-104.

Castano, S., De Antonellis, V., Fugini, M. G., & Pernici, B. (1998). Conceptual schema analysis: techniques and applications. *ACM Transactions on Database Systems* (TODS), 23(3), 286-333.

Castano, S., & De Antonellis, V. (2001). Global viewing of heterogeneous data sources. Knowledge and Data Engineering, IEEE Transactions on, 13(2), 277-297.

Chen, N., He, J., Yang, C., and Wang, C. (2012). A node semantic similarity schema-matching method for multi-version Web Coverage Service retrieval. *International Journal of Geographical Information Science*, 26(6), 1051-1072.

Chua, C. E. H., Chiang, R. H., and Lim, E. P. (2003). Instance-based attribute identification in database integration. *The VLDB Journal*, 12(3), 228-243.

Choi, Youngseok, Jinsoo Park, and Jungsuk Oh (2014), "A Novel Approach to Managing the Dynamic Nature of Semantic Relatedness, Working Paper.

Christiane Fellbaum (1998, ed.) WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press.

Couto, F. M., Silva, M. J., & Coutinho, P. M. (2007). Measuring semantic similarity between Gene Ontology terms. *Data & knowledge engineering*, 61(1), 137-152.

Cover TM, Hart PE (1967). "Nearest neighbor pattern classification". IEEE *Transactions on Information Theory*, 13 (1), 21–27

Cruz, I. F., Antonelli, F. P., and Stroe, C. (2009). AgreementMaker: efficient matching for large real-world schemas and ontologies. *in Proceedings of the VLDB Endowment*, 2(2), 1586-1589.

Dhamankar, R., Lee, Y., Doan, A.H., Halevy, Al, and Domingos, P. (2004). iMAP: Discovering complex semantic matches between database schemas, in Proceedings of the 2004 ACM SIGMOD international conference on Management of data, ACM.

Ding, G., and Wang, G. (2011, January). Discovering implicit categorical semantics for schema matching. In Database Systems for Advanced Applications. Springer Berlin Heidelberg. 179-194

Do, H. H., and Rahm, E. (2002). COMA: a system for flexible combination of schema matching approaches. *in Proceedings of the 28th international conference on the VLDB*, 610-621.

Do, H. H., Melnik, S., and Rahm, E. (2003). Comparison of schema matching evaluations. In Web, Web-Services, and Database Systems. Springer Berlin Heidelberg. 221-237

Do, H.H., Schema Matching and Mapping-based Data Integration. Ph. D. Dissertation, Interdisciplinary Center for Bioinformatics and Department of Computer Science, University of Leipzig, August 2005.

Doan, A., Domingos, P., & Levy, A. Y. (2000, May). Learning Source Description for Data Integration. in WebDB (Informal Proceedings) , 81-86.

Doan, A., Domingos, Pedro, and Halevy, Alon Y (2001). Reconciling schemas of disparate data sources: A machine-learning approach. *ACM SIGMOD Record* 30(2): 509-520.

Doan, A., Madhavan, J., Domingos, P., & Halevy, A. (2002, May). Learning to map between ontologies on the semantic web. In Proceedings of the 11th international conference on World Wide Web (pp. 662-673). ACM.

Domshlak, C., Gal, A., & Roitman, H. (2007). Rank aggregation for automatic

schema matching. *IEEE Transactions on Knowledge and Data Engineering*, 19(4), 538-553.

Drumm, C., Schmitt, M., Do, H. H., and Rahm, E. (2007). Quickmig: automatic schema matching for data migration projects. *in Proceedings of the 16th ACM conference on Conference on information and knowledge management*, 107-116.

Duda, R. O., Hart, P. E., and Stork, D. G. (2012). Pattern classification. John Wiley & Sons.

Embley, D. W., Jackman, D., & Xu, L. (2001, April). Multifaceted Exploitation of Metadata for Attribute Match Discovery in Information Integration. In Workshop on information integration on the Web (pp. 110-117).

Euzenat, Jérôme. (2004) An API for ontology alignment. In *Proceedings of the Semantic Web–ISWC 2004*, 698-712.

Evermann, J. (2009). "Theories of meaning in schema matching: An exploratory study." *Information Systems* 34(1): 28-44.

Fan, W., Li, J., Ma, S., Tang, N., Wu, Y., and Wu, Y. (2010). Graph pattern matching: from intractable to polynomial time. *in Proceedings of the VLDB Endowment*, 3(1-2), 264-275.

Feng, Y., Zhao, L., and Yang, J. (2010, November). GATuner: Tuning Schema Matching Systems Using Genetic Algorithms. In Database Technology and Applications (DBTA), 2010 2nd International Workshop on IEEE.1-4

Gal, Avidgor. (2006). Why is Schema Matching Tough and What Can We Do about It? *SIGMOD Record*, 35(4), 1-11.

Gal, A., Sagi, T., Weidlich, M., Levy, E., Shafran, V., Miklós, Z., & Hung, N. Q. V. (2012, May). Making sense of top-k matchings: A unified match graph for schema matching. In Proceedings of the Ninth International Workshop on ACM Information Integration on the Web. 1-6

Giunchiglia, F., Shvaiko, P., and Yatskevich, M. (2005). Semantic Schema Matching. *in On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, Springer Berlin Heidelberg, 347-365.

Goh, C. H., Bressan, S., Madnick, S., & Siegel, M. (1999). Context interchange: New features and formalisms for the intelligent integration of information. *ACM Transactions on Information Systems* (TOIS), 17(3), 270-293.

Gong, J., Cheng, R., & Cheung, D. W. (2012). Efficient management of uncertainty in XML schema matching. The VLDB Journal, 21(3), 385-409.

Gottlob, G., Pichler, R., & Savenkov, V. (2009). Normalization and optimization of schema mappings. Proceedings of the VLDB Endowment, 2(1), 1102-1113.

Halevy, A. Y., and Madhavan, J. (2003) Corpus-based knowledge representation. In Proceedings of the 18th international joint conference on Artificial intelligence, 1567-1572

Hand, David., Mannila, H., and Smyth, P. (2001). Principles of data mining. MIT press.

He, B., and Chang, K. C. C. (2004). A holistic paradigm for large scale schema matching. *ACM SIGMOD Record*, 33(4), 20-25.

Hernández, M. A., Papotti, P., & Tan, W. C. (2008). Data exchange with data-metadata translations. Proceedings of the VLDB Endowment, 1(1), 260-273.

Hosain, S., & Jamil, H. (2010). Algebraic operator support for semantic data fusion in extended SQL. *in Proceedings of IEEE 9th International Conference on Cybernetic Intelligent Systems (CIS)*, 1-6.

Hull, R. (1997, May). Managing semantic heterogeneity in databases: a theoretical prospective. In Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems. ACM. 51-61

Hyafil, Laurent, Rivest, RL (1976). "Constructing Optimal Binary Decision Trees is NP-complete". *Information Processing Letters* 5 (1), 15–17

Islam, A., Inkpen, D., and Kiringa, I. (2008). Applications of corpus-based semantic similarity and word segmentation to database schema matching. *The VLDB Journal*, 17(5), 1293-1320.

Jaiswal, A., Miller, D. J., and Mitra, P. (2010). Uninterpreted schema matching with embedded value mapping under opaque column names and data values. *IEEE Transactions on Knowledge and Data Engineering*, 22(2), 291-304.

Jaiswal, A., et al. (2013). "Schema matching and embedded value mapping for databases with opaque column names and mixed continuous and discrete-valued data fields." *ACM Transactions on Database Systems,* 38(1), 1-34.

Jannink, J., Mitra, P., Neuhold, E., Pichai, S., Studer, R., Wiederhold, G.(1999) An Algebra for Semantic Interoperation of Semistructured Data. In *Proceedings of 1999 IEEE Knowledge and Data Engineering Exchange Workshop (KDEX'99)*, 77-84.

Jaro, M. A. (1989). "Advances in record linkage methodology as applied to the 1985 census of Tampa Florida". Journal of the American Statistical Association 84 (406): 414–420. doi:10.1080/01621459.1989.10478785.

Jaro, M. A. (1995). "Probabilistic linkage of large public health data file". Statistics in Medicine 14 (5–7): 491–498. doi:10.1002/sim.4780140510. PMID 7792443.

Jeong, B., Lee, D., Cho, H., & Lee, J. (2008). A novel method for measuring semantic similarity for XML schema matching. Expert Systems with Applications, 34(3), 1651-1658.

Kang, J., and Naughton, J. F. (2003) On Schema Matching with Opaque Column Names and Data Values, *in Proceedings of ACM SIGMOD 2003*, 205-216

Kang, J., Lee, D., and Mitra, P. (2005). Identifying value mappings for data integration: An unsupervised approach, *in Proceedings of Conference of Web Information Systems Engineering (WISE 2005)*, Springer Berlin Heidelberg, 544-551.

Kashyap, Vipul. and Amit Sheth (1996). "Smantic and schematic similarity between database objects a context based approach." *VLDB Journal,* 5(4), 276-304.

Kim, J. M., Shin, H., & Kim, H. J. (2007). Schema and constraints-based matching and merging of Topic Maps. *Information processing & management*, 43(4), 930-945.

Kotsiantis, S. B. (2007). Supervised machine learning: a review of classification techniques. *Informatica* , 31(3), 249-268.

Lee, J. H., Kim, M. H., & Lee, Y. J. (1993). Information retrieval based on conceptual distance in IS-A hierarchies. *Journal of documentation*, 49(2), 188-207.

Lee, Y., Sayyadian, M., Doan, A., and Rosenthal, A. S. (2007). eTuner: tuning schema matching software using synthetic scenarios. *The VLDB Journal*, 16(1), 97-122.

Levenshtein, V. I. (1966, February). Binary codes capable of correcting deletions, insertions and reversals. In Soviet physics doklady (Vol. 10, p. 707).

Li, W. S., and Clifton, C. (1994, September). Semantic integration in heterogeneous databases using neural networks. in Proceedings of VLDB Conference 1994, 12-15.

Li, W. S., Clifton, C., and Liu, S. Y. (2000). Database integration using neural networks: Implementation and experiences. *Knowledge and Information Systems*, 2(1), 73-96.

Li, W. S., & Clifton, C. (2000). SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. *Data &*

Knowledge Engineering, 33(1), 49-84.

Lin, D. (1998). An Information-Theoretic Definition of Similarity. in Proceedings of the Fifteenth International Conference on Machine Learning.

Lu, J., Wang, J., & Wang, S. (2007). XML schema matching. *International Journal of Software Engineering and Knowledge Engineering*, 17(05), 575-597.

Madhavan, J., Bernstein, P. A., and Rahm, E. (2001, September). Generic schema matching with Cupid. *in Proceedings of the International Conference on VLDB*, 49-58

Madhavan, J., P. Bernstein, K. Chen, A. Halevy, and P. Shenoy. (2003) Matching schemas by learning from a schema corpus. In Proc. of the IJCAI-03 Workshop on Information Integration.

Madhavan, J., Bernstein, P. A., Doan, A., and Halevy, A. (2005). Corpus-based schema matching. *in Proceedings of IEEE 21st International Conference on Data Engineering (ICDE 2005)*, 57-68.

McCann, R., Shen, W., & Doan, A. (2008, April). Matching schemas in online communities: A web 2.0 approach. In Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on IEEE. 110-119

Melnik, S., Garcia-Molina, H., & Rahm, E. (2002). Similarity flooding: A versatile graph matching algorithm and its application to schema matching. *in Proceedings of 18th IEEE International Conference on Data Engineering*, 117-128.

Mihalcea, R., Corely, C. and Strapparava, C. (2006), Corpus-based and Knowledge-based Measures of Text Semantic Similarity, in Proceedings of AAAI'06 Proceedings of the 21st national conference on Artificial intelligence - Volume 1, 775-780

Miller, R., Ioannidis, Y.E., Ramakrishnan, R. (1994) Schema Equivalence in Hetereogeneous Systems: Bridging Theory and Practice. *Information Systems* 19(1), 3-31.

Milo, T., & Zohar, S. (1998, August). Using schema matching to simplify heterogeneous data translation. in Proceedings of VLDB Conference, New York, U.S., 24-27.

Mitra, P., Wiederhold, G., Kersten, M. (2000). A Graph-Oriented Model for Articulation of Ontology Interdependencies. *In Proceedings of Extending DataBase Technologies, EDBT* LNCS, Springer Verlag

Murthy, S. K. (1998). Automatic construction of decision trees from data: A multi-disciplinary survey. *Data mining and knowledge discovery*, 2(4), 345-389.

Nguyen, T. H., Nguyen, H., & Freire, J. (2010, October). PruSM: a prudent schema matching approach for web forms. In Proceedings of the 19th ACM international conference on Information and knowledge management. ACM. 1385-1388

Nguyen, Q. V. H., Weidlich, M., Nguyen Thanh, T., Gal, A., Aberer, K., & Miklos, Z. (2014). Pay-as-you-go Reconciliation in Schema Matching Networks. In

The 30th IEEE International Conference on Data Engineering (No. EPFL-CONF-189892).

Palopoli, L., Sacca, D., & Ursino, D. (1998, July). Semi-automatic, semantic discovery of properties from database schemes. in Proceedings of International Database Engineering and Applications Symposium, IEEE, 244-253.

Palopoli, L., Terracina, G., & Ursino, D. (2000, September). The System DIKE: Towards the Semi-Automatic Synthesis of Cooperative Information Systems and Data Warehouses. In ADBIS-DASFAA Symposium (pp. 108-117).

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research,* 12, 2825-2830.

Peukert, E., Massmann, S., & Koenig, K. (2010). Comparing Similarity Combination Methods for Schema Matching. GI Jahrestagung (1), 175.

Po, L., and Sorrentino, S. (2011). Automatic generation of probabilistic relationships for improving schema matching. *Information Systems*, 36(2), 192-208.

Quinlan, J. R. (1993). C4. 5: programs for machine learning (Vol. 1). Morgan kaufmann.

Rada,R., E. Bicknell (1989), Ranking documents with a thesaurus. *Journal of the American Society for Information Science*, 40 (5), p.304-310.

Rada, R., Mili, H., Bicknell, E., & Blettner, M. (1989). Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man and Cybernetics,* 19(1), 17-30.

Rahm, E. and Bernstein, P. A. (2001). A Survey of Approaches to Automatic Schema Matching. *The VLDB Journal*, 10(4), 334-350.

Rahm, E. (2011). Towards large-scale schema and ontology matching. In Schema matching and mapping, Springer Berlin Heidelberg, 3-27.

Resnik, P. (1995). Using Information Content to Evauate Semantic Similarity in a Taxonomy. in Proceedings of the 14th International Joint Conference on Artificial Intelligence,Montreal, Quebec, Canada

Sabetzadeh, M., and Easterbrook, S. (2005). An algebraic framework for merging incomplete and inconsistent views. *In Proceedings. 13th IEEE International Conference on Requirements Engineering,* 306-315.

Shvaiko, P. and Jérôme Euzenat. (2005). A survey of schema-based matching approaches. *Journal on Data Semantics IV*, 146-171.

Smiljanić, M., Van Keulen, M., and Jonker, W. (2005). Formalizing the XML schema matching problem as a constraint optimization problem. *in Proceedings of Database and Expert Systems Applications*, Springer Berlin Heidelberg. 333-342.

Smiljanić, M., Van Keulen, M., and Jonker, W. (2006). Using element clustering to increase the efficiency of xml schema matching. *in Proceedings. 22nd IEEE International Conference on Data Engineering Workshops*, 45-54

Thang, H. Q., and Nam, V. S. (2010). XML schema automatic matching solution. *International Journal of Electrical, Computer, and Systems Engineering*, 4(1), 68-74.

Turney, P. (2001, September 3–7, 2001). Mining the Web for Synonyms: PMI-IR Versus LSA on TOEFL. Paper presented at the Twelth European Conference on Machine Learning (ECML-2001), Freiburg, Germany.

Van Dongen, S. (2000) Graph Clustering by Flow Simulation. Ph. D. Thesis, University of Utrecht, The Netherland.

Van Rossum, G. (2010). Python 2.7 Documentation.

Winkler, W. E. (1990). "String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage". Proceedings of the Section on Survey Research Methods (American Statistical Association): 354–359.

Wang, J., Wen, J. R., Lochovsky, F., and Ma, W. Y. (2004). Instance-based schema matching for web databases by domain-specific query probing. *in Proceedings of the Thirtieth international conference on VLDB Endowment*, 30, 408-419.

Yu, C., and Popa, L. (2004). Constraint-based XML query rewriting for data integration. *in Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, 371-382.

Zhang, Z., Che, H., Shi, P., Sun, Y., and Gu, J. (2005). An algebraic framework for schema matching. *in Proceedings of 6th International conference on Web-Age Information Management*, Springer Berlin Heidelberg, 694-699.

Zhang, Z., Che, H., Shi, P., and Gu, J. (2008). An algebraic framework for schema matching. *Informatica*, 19(3), 421-446.

Zhao, C., Shen, D., Kou, Y., Nie, T., and Yu, G. (2012). A Multilayer Method of Schema Matching Based on Semantic and Functional Dependencies. *in Proceedings of IEEE 9th Web Information Systems and Applications Conference (WISA)*, 223-228.

Zhao, H. and S. Ram (2007). "Combining schema and instance information for integrating heterogeneous data sources." *Data & Knowledge Engineering* 61(2), 281-303.

ABSTRACT (KOREAN)

Toward Efficient and Accurate Schema Matching - Cross Similarity Vector Approach and Learning-based Matcher Combination

崔 榮 碩

서울大學校 大學院

經營學科 經營學 博士

IT 시스템이 기업의 필수 역량으로 자리잡은 이후로, IT 시스템 간 통합 및 상호 운용성의 확보는 학계는 물론 실무에서도 중요한 문제로 다루어져 왔다. 기업의 IT관련 지출의 대부분이 시스템 통합비용으로 소모되고 있을 뿐만 아니라, 기업간 인수 합병의 일반화, 데이터원의 다양화로 인해 자동화된 시스템 통합 방법론의 개발 수요가 꾸준히 증가하고 있다. 더욱이 빅데이터 시대의 도래에 따라 기업이 중복 데이터 및 유사 데이터를 효율적으로 관리하는 것이 기업의 핵심 역량으로 떠오르고 있다. 이러한 가운데, 시스템 통합의 기초가 되는

데이터 스키마 매칭(Schema Matching)은 수십여 년 전부터 연구되어왔음에도 불구하고 여전히 개선의 여지가 상당한 실정이다.

스키마 매칭은 그 활용의 범용성에 있어 중요성이 매우 크다고 할 수 있다. 단일 기업 내에서도 데이터 웨어하우스 운영을 위해서는 다양한 출처에서 발생하는 데이터들을 단일화된 시각으로 다루어야 할 뿐만 아니라, 복수 기업의 인수 및 합병 상황에서도 데이터를 하나의 관점에서 관리할 필요성이 생기게 된다. 이러한 다양한 데이터 통합을 위해서는 통합에 앞서 데이터 스키마 매칭이 필수적이다. 스키마 매칭은 통합하고자 하는 두 데이터원의 스키마간의 연결을 찾는 방법론으로, 다양한 방법을 활용해 동일한 스키마 요소를 찾아내게 된다. 뿐만 아니라, 데이터베이스의 질의어 처리에 있어서도 복수의 데이터베이스에서 동일한 데이터 필드가 무엇인지에 대한 정보가 없는 경우 질의어에 대한 원하는 답을 알아내가 어렵다. 사전에 이종 데이터들이 지칭하고 있는 동일한 데이터가 무엇인지에 대한 정보가 있다면 이러한 일들을 보다 쉽게 처리할 수 있게 될 것이다.

데이터의 스키마 매칭을 위해서 기존 연구들은 다양한 방법을 적용해왔다. 컴퓨터 언어학을 기반으로 스키마 요소의 이름 (schema element name)간의 유사성을 계산하는 다양한 방법들이 제시되었을 뿐만 아니라, 데이터 스키마가 갖는 구조적인 유사성을 기반으로 통합을 필요로 하는 두 데이터 스키마를 매칭하는 연구도 제안되었다. 이런

다양한 매칭 방법들이 제시됨에 따라 기존의 매칭 방법들을 조합해 보다 나은 매칭 성과를 내기 위한 노력도 많은 연구자들이 진행해왔다. 스키마 매칭 문제가 갖는 상황의 특수성으로 인해, 단일 매칭 알고리즘이 모든 매칭 상황을 효율적으로 해결할 수 없기 때문에 대부분의 최근 연구들은 기존의 매칭 방법의 조합 방법에 초점을 맞추고 있는 것이 현실이다. 그러나, 대부분의 연구들은 데이터 스키마 이름이 가지고 있는 특수성으로 인해 완벽히 자동화된 방법론을 제안하는데 실패하였다. 데이터 스키마의 이름은 스키마를 디자인하는 디자이너의 개인적인 습성에 따라 약어를 사용하거나 띄어쓰기 등을 생략하는 등 이름의 명시성을 확보하기가 어려운 것이 사실이다. 따라서 이런 이름이 내포하고 있는 의미를 사람이 직접 풀어내는 과정을 채택하여 전체 매칭 시스템의 효율성을 높여왔다. 그러나, 현실적으로 사람이 수동으로 스키마 이름이 내포하고 있는 의미를 풀어내는 과정을 진행하는 것은 매우 어려운 일일 뿐만 아니라, 자동화를 위해서는 반드시 풀어야 하는 문제이다. 또한 단일 매칭 방법들이 수백 여가지가 넘게 제시된 상황에서, 이를 효과적으로 조합해내는 방법론을 찾기는 쉽지 않다. 앞으로도 제시될 많은 단일 매칭 방법론들을 효과적으로 첨가하고, 상황에 맞게 필요한 매칭 방법들만을 활용해 스키마 매칭을 수행하는 방법론이 필요하다고 할 수 있겠다.

단일 매칭 방법론 중 구조적 유사성을 이용해 매칭되는 스키마

요소들을 찾는 방법론도 근원적으로 가지고 있는 단점으로 인해 그 효과성이 떨어지고 있다. 구조적인 유사도 측정 방식의 경우, 스키마가 가지고 있는 구조적 유사성만을 살피게 되어 동일한 의미임에도 불구하고 구조적 디자인이 달리 되어있다는 이유만으로 매칭에 실패하는 경우가 발생하게 된다. 단순히 기하적인 구조만을 살펴 스키마 매칭을 실행하기 때문에 발생하는 문제로 볼 수 있을 것이다.

본 연구에서는, 먼저 기존의 구조적인 유사성을 기반으로 하는 스키마 매칭 방법이 갖는 단점을 보완할 수 있는 Cross Similarity Vector 접근 방법을 제안하였다. 데이터베이스 연구에서 오랜 전통을 가지고 있는 Context의 개념을 도입하여 스키마 요인들의 의미적 Context를 반영하여 매칭을 실행하는 새로운 방법론을 제시하였다. 이 방법론은 기존에 존재하는 의미 기반의 스키마 매칭 방법을 활용하여 구조적인 유사도를 계산함으로써 새로운 유사도 계산 방법을 추가적으로 확보할 수 있을 뿐만 아니라 매칭의 정확도 또한 기존의 구조적 유사성을 기반으로 한 매칭 방법에 비해 높다는 장점을 가지고 있다.

다음으로는 기존에 존재하는 다양한 매칭 방법들을 기계학습 방법론을 기반으로 조합하는 방법론을 제시하였다. 다양한 매칭 방법들을 조합해 내기 위해 기계학습 기반의 분류기를 만들어 소스와 타겟이 되는 스키마간의 유사도를 계산하였다. 사용되는 개별 매칭 방법론들은 매칭을 위한 특성 벡터 공간을 형성하게 되며, 각 방법론을

통해 계산된 유사도 들은 벡터공간상의 점으로 표현되게 된다. 이런 기계학습 기반의 분류기법을 복수의 스키마 매칭 방법의 조합에 적용함으로써, 매칭의 효율성을 확보했을 뿐만 아니라, 매칭의 전 과정을 자동화하는데 성공했다. 스키마 요소의 이름이 갖는 유사성을 찾기 위해 다양한 유사도 기법들을 활용하였고, 약어의 매칭을 위해 Edit distance기반의 다양한 유사도 계산법들이 활용되었다.

본 연구는, 구조적인 유사성을 계산하는 효율적인 방법론을 제안해 냈을 뿐만 아니라, 기존의 많은 스키마 매칭 알고리즘들을 활용할 수 있는 통합적인 프레임워크를 제공했다는 점에서 그 기여점이 크다고 할 수 있다. 더욱이 반자동화된 스키마 매칭 방법론에 비교할 때 본 연구에서 제안하는 자동화 방법이 동일한 수준의 매칭 성능을 보여주고 있기 때문에, 차후 상용화의 가능성도 매우 크다고 할 수 있겠다.

키워드: schema matching, data matching, combinational matching, matcher combination, cross similarity vector, learning-based classification, data integration, machine learning.

학 번: 2008-22826

Youngseok Choi, Ph. D.

Curriculum Vitae

Seoul National University

1 Gwanak-ro Gwanak-gu, Seoul, Korea

Phone: +82-10-5474-3707 / Email: aquinas9@snu.ac.kr

EDUCATION

- ***Seoul National University,*** *SEOUL, KOREA*

Ph.D., Information Systems Management, SNU Business School
2008.8 – 2014. 6

(Master-Ph.D Integrated course of Management Information Systems)

- ***Seoul National University,*** *SEOUL, KOREA*

Bachelor of Electrical Engineering / Management of Technology (Double Major)
2002 – 2008

- Thesis1: Implementing the Systems for Detecting Specific Motion in Video (Electrical Engineering)
- Thesis2: The Impact of Organizational Technology on Organizational Structure (Management of Technology)

DOCTORAL DISSERTATION

- Title: "Toward Efficient and Accurate Schema Matching - Cross Similarity Vector Approach and Learning-based Matcher Combination"
- Dissertation Advisor: **Dr. Jinsoo Park** (Associate Dean of SNU Business School, Seoul National University)

HONORS AND AWARDS

- Doctoral Consortium Fellow, Asian Pacific Conference on Information Systems (PACIS) 2013, Jeju, Korea, June 2013
- SNU Full Scholarship, Seoul National University, 2003
- SNU Scholarship, Seoul National University, 2002, 2004, 2005, 2009

RESEARCH INTEREST

- Data Integration
- Schema Matching
- Machine Learning and Data Mining
- Semantic Web and Ontology Engineering
- Big Data Analysis
- Computational Linguistics
- Analytical Modeling using Random Process and Variable
- Text Mining and Opinion Mining for E-Commerce Environment
- Statistical Analysis
- Semantics

PUBLICATION

Journal

- Jinsoo Park and **Youngseok Choi**, "Economic Recession and Hyundai Heavy Industries - Strategies to cope with the risk factors of heavy industry", **Korea Business Review,** forthcoming

- Jinsoo Park and **Youngseok Choi**, "The Ecosystem of the Smartphone Industry in Korea: Perspectives on Its Sustainable Growth," **Information Systems Review**, Vol. 15, No. 1, April 2013

- **Youngseok Choi** and Jinsoo Park, "The Need for Paradigm Shift in Semantic Similarity and Semantic Relatedness: From Cognitive Semantics Perspective," **Journal of Intelligence and Information Systems** (fast track paper of KIISS Fall Conference 2012, journal version paper),Vol. 19, No. 1, March 2013, pp. 111-123.

- Hyunmi Baek, JoongHo Ahn, and **Youngseok Choi** "Helpfulness of Online Consumer Reviews: Readers' Objectives and Review Cues," **International Journal of Electronic Commerce**, Vol. 17, N0. 2, Winter 2012-13, pp. 99-126.

- Jinsoo Park, Namwon Kim, Minjung Choi, Zhe Jin, and **Youngseok Choi**, "Semantic Search: A Survey," **Journal of Intelligence and Information Systems**, Vol. 17, No. 4, Dec. 2011, pp. 19-36

- Jinsoo Park and **Youngseok Choi** "A Strategy for Discovering New Growth Engine through Innovation: A Case of Hyundai Mobis," **Korea Business Review**, Vol. 15, No. 1, May. 2011, pp. 1-27

- Jinsoo Park, **Youngseok Choi**, "Forecasting Competition of Telecommunication Company in Full Browsing Service Market Based on First-Mover Advantage Analysis," **Information Systems Review**, Vol. 12, No. 1, Apr. 2010, pp. 1-20

Conference

- **Youngseok Choi**, "DYMS(Dynamic Matcher Selector): Scenario-based Matcher Selector,"in Proceedings of PACIS 2013 (Doctoral Consortium), June 18-19, 2013, Jeju Island, Korea.

- Jinsoo Park, Jihae Suh, and **Youngseok Choi**, "Infocuration System Can Save Your Time in Social Networks," in Proceedings of the Post-ICIS 2012 LG CNS/KrAIS Workshop, Orlando, Florida, Dec. 2012

- **Youngseok Choi** and Jinsoo Park, "The Need for Paradigm Shift in Semantic Similarity and Semantic Relatedness: From Cognitive Semantics Perspective," in Proceedings of the 2012 KIISS Fall Conference, Seoul, Korea, December 7, 2012, pp. 212-217.

- **Youngseok Choi** and Jinsoo Park, "A Method for Constructing Semantic Network and Computing Dynamic Semantic Relatedness based on Web Community Corpus Analysis," in Proceedings of the 2012 KIISS Spring Conference, pp. 125-131, Seoul, Korea, May. 2012

COMPLETE RESEARCH

- **Youngseok Choi**, Jinsoo Pakr, and Jungsuk Oh, " A Novel Approach to Managing the Dynamic Nature of Semantic Relatedness , " **Under Review**, ***Data and Knowledge Engineering***

- **Youngseok Choi** and Jinsoo Park, "Schema Matching for Data Integration: Learning-based Matcher Selection"

- **Youngseok Choi** and Jinsoo Park, "Combining Structural and Semantic Measure for Schema Matching: Cross Similarity Vector Approach"

- **Youngseok Choi, "Factor affecting the performance of sentiment analysis techniques – Its comparative analysis"**

WORKS in PROGRESS

- **Youngseok Choi**, "Sentimental Analysis: Voting Ensemble-based Approach"

- **Youngseok Choi,** "Schema Graph Matching: Convex Optimization-based Approach"

- **Youngseok Choi** , "Modeling the Information Cascades and Diffusion in Social Broadcasting Media"

- **Youngseok Choi** and Jinsoo Park**,** "DocuOnto: Ontology Ranking System"

TEACHING EXPERIENCE

- Management Information Systems, Fall 2013, Business School, Seoul National University (Lectured in English)

- Management Information Systems, Spring 2013, College of Aviation and Management, Korea Aerospace University

- Management Information Systems, Fall 2012, Business School, Seoul National University

RESEARCH PROJECT

- **Researcher** in AACSB Accreditation Project – AoL (Assurance of Learning) Researcher, SNU Business School (2014)

- **Researcher** in QoLT Project (2011, Center for Quality of Life Technology, Seoul National University)

- **Main Researcher** in Project for Analysis of U.S. NTIS service, (2010, Korea Institute of Science and Technology Information)

INDUSTRIAL EXPERIENCE

- Research Team Manager - Softbridge (Investment Company), Seoul, Korea **July 2005 – February 2007**

  - Development of System Trading Algorithm using Stochastic Process
  - Development of System Trading Application for General Customer
  - Stock/ Future Price Forecasting System

- Team Member of R&D Department - Biospace, Seoul, Korea **September 2004 – July 2005**

  - Short-term project member for manufacturing technology of Body Composition Analyzer
  - Project member for development of Automatic Height Measurement device

SKILLS

- Proficient in statistical analysis and software (R, SPSS, SAS)

- Proficient in programming using C/C++, Java, Python, Delphi, and Borland C++ Builder.

- Proficient in network analysis using UCINET, Pajec, and NetMiner

- Proficient in linked data handling using Protégé (owl/rdf)

LANGUAGE

- Korean (Mother tongue), English (Fluent in Writing and Speaking)

INVITED TALKS AND PRESENTATIN

- "Strategy to Capture and Create Value from Public Sector Big Data," presented in Korea Workers' Compensation and Welfare Service, Seoul Branch, January 23, 2014

- "How Big Data Creates Jobs ," presented in Ministry of Employment and Labor, Seoul Branch, October 2, 2013