ISBN 978-82-326-4842-9 (printed ver.)
ISBN 978-82-326-4843-6 (electronic ver.)
ISSN 1503-8181

NTNU
Norwegian University of
Science and Technology

Audun Vennesland

Doctoral theses at NTNU, 2020:247

NTNU
Norwegian University of Science and Technology
Thesis for the Degree of
Philosophiae Doctor
Faculty of Information Technology and Electrical
Engineering
Department of Computer Science

NTNU

Doctoral thesis

Doctoral theses at NTNU, 2020:247

Audun Vennesland

Semantic Matching

Dynamic Composition of Matcher Ensembles for Ontology Alignment

NTNU
Norwegian University of
Science and Technology

Audun Vennesland

Semantic Matching

Dynamic Composition of Matcher Ensembles for Ontology Alignment

Thesis for the Degree of Philosophiae Doctor

Trondheim, September 2020

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the Degree of Philosophiae Doctor

Faculty of Information Technology and Electrical Engineering
Department of Computer Science

ISBN 978-82-326-4842-9 (printed ver.)
ISBN 978-82-326-4843-6 (electronic ver.)
ISSN 1503-8181

Doctoral theses at NTNU, 2020:247

Printed by NTNU Grafisk senter

Abstract

Semantic matching is a computational process that aims to automatically identify the semantic relationship between elements represented in different graph-like sources. Typically, this is a process that involves human decision-making when it comes to selecting appropriate matching algorithms, configuring their similarity thresholds, and aggregating their results into a final alignment. This thesis proposes a more autonomous approach where such decisions are automatically determined by an analysis of terminological, structural and lexical features extracted from the ontologies to be matched.

A design science-centred research approach has guided the development. In several build-and-evaluate loops, matching artefacts have been developed and evaluated, iteratively improving the artefacts themselves as well as extracting lessons learned that extend current knowledge. The produced artefacts encompass *ontology profiling metrics* that capture relevant features of the ontologies to be matched, *matching algorithms* that automatically compute an alignment holding equivalence and subsumption relations between concepts of two input ontologies, an *alignment combination method* that optimally combines the results from the ensemble of algorithms, and *mismatch detection techniques* that filter out false positive relations caused by ontology mismatches or heterogeneities. These individual artefacts are finally combined into a prototype semantic matching system.

The individual matching artefacts as well as the prototype system have been evaluated in three diverse datasets. In general, the evaluation results show that the proposed approach improves the quality of individual alignments as well as the combined alignment. Furthermore, the results confirm that some of the new ideas implemented in the matching algorithms contribute to the identification of "challenging" relations and that the suggested mismatch detection techniques can increase alignment precision.

Keywords
Semantic Matching, Ontology Matching, Semantic Interoperability, Semantic Web, Data Integration, Ontology Engineering

To the memory of my father
Øystein Vennesland
October 8th 1942 – November 28th 2019

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) for the partial fulfilment of the requirements for the degree of Philosophiae Doctor.

This doctoral work has been conducted at the Data and Artificial Intelligence group (DART), Department of Computer Science (IDI), Faculty of Information Technology and Electrical Engineering (IE). The work has been performed under the supervision of Associate Professor Trond Aalberg. Professor Heri Ramampiaro and Professor Jon Atle Gulla were assigned as co-supervisors.

Acknowledgements

Many people have contributed to shape the work summarised in this thesis, but I want to start by saying how much I value the support from my closest family. Tone, my significant other, has been unimaginably patient, and has offered understanding and care, both in good PhD times and bad. My two kids, Gabriel and Jesper, have helped me keep focus on the important things in life and regain energy after having spent way too many hours in front of the computer. I'm also very grateful to my parents, Gunhild and Øystein, for giving me a solid foundation in my upbringing.

A great thanks to Dr. Trond Aalberg, my supervisor, who's always been willing to offer guidance, but at the same time has given me freedom to choose my own directions during this work.

From SINTEF, my employer, a number of people have offered their support during this work. A special thanks to Eldfrid Øvstedal, who made it possible to take a break from my "day job" at SINTEF to pursue a PhD; Marit Natvig, who during our many years working together has motivated my interest for interoperability challenges; Ivonne Herrera, for nice conversations about work as well as the finer things in life (such as good coffee and food); Joe Gorman, for helping me position semantic matching in the air traffic management domain and for nice chats about our memories from Spain; Ståle Walderhaug and Per Gunnar Auran for encouraging advice and for offering valuable comments to this dissertation.

During these years I've been very fortunate to meet and collaborate with many friendly and incredibly clever people: Yoan Gutierrez, my local contact and friend at the University of Alicante; Giulio Petrucci (Google), for insightful discussions on how word embeddings can contribute to semantic matching; Fabien Duchateau from the University of Lyon, who with his experiences from schema and ontology matching has been a good discussion partner and helped me scope my work; and Christoph Schuetz (Johannes Kepler University), Bernd Neumayr (Johannes Kepler University), Eduard Gringinger (Frequentis), Rich Keller (NASA) and Scott Wilson (Eurocontrol) for interesting collaborations related to ontologies in air traffic management.

Thank you all.

Contents

List of Figures

List of Tables

Part I

Background and Context

1 Introduction

1.1 Motivation and Problem Outline

Semantic matching refers to a process where the relations between semantically corresponding nodes from two graph-like structures are discovered by computing [47]. Several application areas, such as data- and information integration and information retrieval, rely on the ability to automatically or semi-automatically identify semantic relations among structured models such as ontologies, schemas, taxonomies or vocabularies. Semantic matching, which encompasses research areas such as schema matching [9], ontology matching [35], taxonomy matching [6], and semantic matchmaking [4, 73, 128], aims to identify different semantic relations between heterogeneous sources using a variety of automated or semi-automated techniques.

Schema and ontology matching have been active research areas for several decades, and over time new sub-research areas have also emerged, such as large-scale matching; user involvement in matching; social and collaborative matching; benchmarking and evaluation of matching systems; and alignment management infrastructure and support, to name only a few [35]. Furthermore, several different research disciplines are involved, primarily computer science, but also mathematics, engineering, social sciences, business- and management and psychology. A result of this extensive research is that a large number of matching systems and techniques have been developed and the performance of such matching systems has improved significantly over the years. And the research field is still very active.

At present, as ontology engineering is transitioning from basic to applied research and is becoming a more commonplace activity, a mass of new ontologies originate from application domains that traditionally have not been a part of the semantic web arena. This imposes new levels of complexity that upholds the momentum of the semantic matching research area. More recent developments and studies in this area focus on the identification of semantic relations beyond 1-1 class equivalence, such as property matching [18], subsumption matching [79], and complex matching [147]. This direction is also recognised by the Ontology Alignment Evaluation Initiative (OAEI), an annual benchmarking campaign for matching systems. In 2018, the task of computing semantic relations beyond 1-1 equivalence was again[1] put back on the agenda when the "Complex Matching" track was arranged.

Automated identification of semantic relations is a challenging task due to different types of heterogeneities or mismatches that exist among the ontologies to be matched. A general view is that the task of automatically identifying semantic relations between ontologies can never be fully automated. There will always be different conceptual and explication heterogeneities that require some form of human intervention. This view is certainly also shared in this work, but the assumption is that there is still significant improvement to be made, both with respect to increasing the level of automation and the scope of functionality for such systems.

In the following, we highlight three inter-related areas where state-of-the-art can be extended and that represent the core of this thesis.

1.1.1 Identification of semantic relations beyond equivalence

Most current ontology matching systems focus on class equivalence matching, while other semantic relations between the sources to be integrated are largely neglected, despite being considered an important prerequisite for a more holistic integration approach [22, 137]. Although a list of equivalent elements is helpful, it is only a starting point for a more profound integration process, where also asymmetric relations such as subsumption and meronomy need to be considered. Especially when the ontologies to be matched have different granularity levels or represent partly overlapping scopes, which is often the case, the identification of such asymmetric semantic relations is particularly useful [137, 22]. Furthermore, their identification can also inform discovery of additional correct equivalence relations

[1] In 2011, OAEI arranged a track called Oriented Matching that challenged systems capable of identifying subsumption relations. This is the same OAEI dataset that is used in the evaluation in this thesis.

as well as removal of incorrect ones during the matching process.

One reason why subsumption matching lags behind equivalence matching is the lack of benchmarks for systems and techniques targeting such relations [162]. A contribution from this work is the development of two new datasets that can be used to evaluate techniques for detecting both equivalence and subsumption relations. These datasets represent different application domains and have different size and complexity. Together with a dataset from the OAEI, these two datasets are used to evaluate the different artefacts developed in this work. Evaluating the suggested approach in three such diverse datasets supports generalisability and helps avoid overfitting the techniques to a particular context.

1.1.2 Automated matcher selection, matcher configuration and alignment combination

Due to the diversity of (mostly) humanly engineered models such as ontologies, a single matching algorithm will rarely produce a good alignment on its own [35, 120, 93]. The matching process is therefore normally approached using an ensemble of matchers or matching algorithms [87]. In such a setup, each matcher computes a set of relations based on a certain target characteristic of the ontologies to be matched. Usually, the composition of the different matchers and their configuration is performed manually, not only by proficient ontology matching system users but also by domain experts and ontology engineers. However, configuring and tuning such a system, with many matchers, combination methods, and individual parameter settings, is a task far from trivial, even for experts [93, 56]. Moreover, even if an ensemble of matching algorithms is employed, you cannot run the same ensemble of matching algorithms, with the same configuration, for any pair of ontologies to be matched, as semantic matching is a highly context-dependent process. In sum, this is a comprehensive effort that could be alleviated by automated means and this thesis sets out to develop an approach for making the matching process more autonomous. In the suggested matching process, matchers are configured and orchestrated automatically from an analysis of the profile of the ontologies to be matched as well as capabilities of the available matchers. The overall approach involves three sub-processes:

1. Perform an analysis of the terminological, structural and lexical characteristics of the ontologies to be matched to establish a set of profiling metrics of the ontologies.

2. Select and configure appropriate equivalence and subsumption matchers by applying the profiling metrics captured in (1).

3. Combine the alignments from the selected matchers to produce an optimal final alignment.

1.1.3 Dealing with ontology mismatches

Ontologies to be aligned often include different types of mismatches, also called heterogeneities [35], caused by different conceptualisations of the domain, different development principles and patterns, differing scopes and underlying standards, different terminology, to name a few. In particular, this is a precision problem as such mismatches can result in false positive relations being added to an alignment when the mismatches are not detected by the matchers. In this work, we review literature related to ontology mismatches and try to derive heuristics that can be used for automatically detecting mismatched relations in the post-matching phase. In other words, the mismatch detection strategies aim to improve the precision of the produced alignments by filtering out false positive relations contributed by mismatches.

Furthermore, most matching systems largely rely on some form of syntactic processing of ontology concept names using one or more string matching techniques [15]. String matching techniques have the advantage of being fast and as long as the syntactic equality reflects the semantic equality, these techniques often yield good results. However, the heterogeneities or mismatches mentioned above call for a more profound analysis of the ontology concepts than basic string matching algorithms are capable of performing. In the matcher ensemble used in this work string matching techniques are replaced by techniques that exploit word embeddings, i.e. words from the corpus are "embedded" in a vector space. The word-to-vector representation is based on a semantic analysis since the vectors are a result of a learning process that, among other aspects, takes into account how a given word relates to other words in its context. Hence, these embeddings act as semantic proxies from which semantic relations between words are deduced, rather than analysing the local structures (i.e. characters) of the words to be compared.

1.2 Objectives and Research Questions

The main objective of this research is to:

Develop an approach for semantic matching that uses inherent characteristics of ontological models to produce an alignment that includes both equivalence and subsumption relations.

This overall objective encompasses the following sub-objectives:

- Identify metrics that quantitatively define profiles of ontologies to be matched and that can further be used to select, configure and combine a set of matching algorithms.
- Develop and evaluate matchers producing both equivalence and subsumption relations between concepts of heterogeneous ontologies.
- Identify strategies for selecting and configuring the most relevant matchers based on ontology profiling metrics.
- Identify strategies that in an optimal manner combine the alignments produced by the relevant matchers based on ontology profiling metrics.
- Identify strategies for detecting ontology mismatches in order to enhance the final alignment returned by the matching process.
- Develop and evaluate a proof-of-concept prototype of a semantic matching system that integrates all artefacts emerging from the above sub-objectives.

Based on the above objectives the following research questions have been defined:

RQ1: Which ontology characteristics can guide the composition of a relevant ensemble of matchers in a semantic matching system?
To automatically select a set of appropriate matching algorithms the system includes a set of profiling metrics that quantifies and analyses different characteristics of the ontologies to be matched. In the ontology evaluation literature, there is a vast amount of metrics that extract quantitative characteristics related to the terminological, structural, and linguistic properties embedded in ontologies. The position of this work is that these characteristics can be employed to select a set of optimal matchers from a library of matchers. Furthermore, once an optimal set of matchers has been appointed for a given matching task, the matchers have to be configured and the alignments they produce will have to be combined to return an as optimal final alignment as possible. Different matchers all have their strengths and weaknesses, they focus on different perspectives of the ontologies to be

matched, and the objective is to have an as complementary set of matchers as possible. To accomplish this, the matchers have to be tuned with respect to the confidence assigned to their similarity measurements and how much weight each matcher should be given when run in an ensemble together with other matchers.

RQ2: Which techniques can be used to automatically identify subsumption relations?
A wide range of techniques has been proposed for the automatic identification of equivalence relations. However, when asymmetric relations, such as subsumption relations, are to be inferred, different techniques are needed. Although some work has been done in this area before (e.g. by Giunchiglia et al. [46] and Arnold and Rahm [7]), it is quite limited compared to that of equivalence matching. It is therefore assumed that more concentrated research on subsumption matching can help advance state of the art in semantic matching.

RQ3: Which combination strategies are applicable when combining semantic relations - produced by an ensemble of equivalence and subsumption matchers - into a final alignment?
When equivalence alignments are combined this is often based on the "single marriage" principle, that is, there should be a 1-1 relation between the best matching relation between two ontology concepts from different ontologies. Such an approach will clearly not work for subsumption relations, since a concept in one ontology is likely related to several concepts in the other ontology, and vice-versa. Investigating candidate combination methods and evaluating how they perform will shed light on an important component of a semantic matching system, namely how do we aggregate the best quality equivalence and subsumption relations from individual alignments while disregarding those that reduce the quality.

RQ4: Which strategies can be used to automatically detect ontology mismatches and ultimately enhance the quality of already produced alignments?
The quality of the alignment returned from a semantic matching system is measured by how many correct relations the system is able to identify and how many false relations the system can avoid. This research question relates to different techniques that contribute to the latter. In order to address it, this work will, supported by existing knowledge on ontology mismatches, investigate techniques that can be used to filter out false positive relations computed by the semantic matching system.

1.3 Research Method

The guiding research framework used in this work is based on Design Science [65, 64]. Design Science prescribes build-and-evaluate loops where artefacts are developed through iterative and rigorous evaluation using empirical evaluation methods. Gold-standard evaluation using evaluating metrics typically applied in the ontology matching community is used to evaluate the artefacts developed. Part II describes the research approach for this work more in detail.

1.4 Major Contributions

This work extends the knowledge base within the area of semantic matching with the following core contributions:

- Ontology profiling metrics that define ontology characteristics used for selecting the optimal set of matchers as well as their configuration and combination.
- A set of matching algorithms that automatically identify equivalence and subsumption relations.
- A strategy for employing the ontology profiling metrics into a weighted combination of the alignments produced by the individual matching algorithms.
- Two mismatch detection techniques that contribute to remove false positive relations and consequently increase the precision of alignments produced by the matching algorithms without suffering recall.

In addition, this research has produced two datasets that can be used to evaluate equivalence and subsumption matching algorithms and systems.

8 papers have been produced during this work and they are all described in Appendix A.

1.5 Thesis Structure

The remainder of the thesis is structured as follows.

Part I Background and Context.
Chapter 2 introduces some basic concepts relevant to this thesis. The

chapter begins with a short introduction to ontologies before the fundamentals of semantic matching are explained. Further, this chapter presents the different semantic relations that are most relevant in this work, as well as various techniques that can be used for their identification.

Chapter 3 starts by presenting an overview of existing approaches for automatically detecting equivalence and subsumption relations. Next, this chapter describes relevant research related to extracting and measuring ontology characteristics that can be used for ontology profiling, as well as different approaches related to matcher selection, matcher configuration, and combination of matcher results.

Part III Research Approach.
Chapter 4 first gives an introduction to the Design Science framework, and then explains how this framework has guided the development-oriented research in this thesis.

Part IV Implementation and Evaluation.
Chapter 5 describes the development of the different artefacts that together compose a prototype of a semantic matching system. These artefacts include ontology profiling metrics, equivalence matchers, subsumption matchers, alignment combination strategies and mismatch detection strategies.

Chapter 6 describes the evaluation of the developed artefacts in three diverse datasets.

The most significant results from the evaluation along with a discussion about the validity, reliability and credibility of the research are presented in Chapter 7.

Part VI Conclusions and Further Work.
Chapter 8 summarises the main conclusions, the most important contributions from this work and how they address the research questions, before it concludes with some ideas for further work.

2 Background and Preliminaries

2.1 Ontologies

This section provides a minimal and practical description of some key aspects related to ontologies to prepare for the remainder of this thesis. For a more detailed explanation of ontologies and their application, the reader is referred to the "Handbook on Ontologies" [138].

An ontology is a formal definition of the concepts, properties and interrelationships of the entities that exist in some domain of discourse. It provides a shared vocabulary that can be used to describe the domain, classifying and categorising the elements contained within it.

Typically, an ontology is formalised using the Web Ontology Language (OWL)[1]. OWL is a part of the W3C suite of Semantic Web standards[2], which includes among others Resource Description Format (RDF)[3], a framework for representing web data using subject-predicate-object triples, and the Resource Description Format Schema (RDFS)[4], which provides a data-modelling vocabulary for RDF data. While both OWL and RDFS offer a vocabulary for describing RDF data, OWL allows for greater expressibility than RDFS.

In an ontology, classes represent sets of individuals (also called instances or objects) with similar characteristics and are organised in an specialisation

[1] https://www.w3.org/TR/owl2-overview/

[2] https://www.w3.org/standards/semanticweb/

[3] https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/

[4] https://www.w3.org/TR/rdf-schema/

hierarchy. This hierarchy is also called a *subsumption hierarchy* in that a parent class subsumes its children classes, i.e. any individual that is a member of the subsumed (more specific) class is also a member of its subsuming (more general) class. Figure 2.1 shows the extract of an ontology where its concepts describe different aspects of a doctoral thesis. The example includes classes (rectangles), object properties and data properties (ovals), data types (hexagons), and individuals (in braces). For example, a particular doctoral thesis being a member of the *PhDThesis* class is also a member of the *Thesis* class. Object properties relate one instance to another instance. *author* is an example of an object property that relates individuals in the *Thesis* class to individuals in the *Person* class. Data properties map individuals to literals (such as how the language property states that an individual of the *Thesis* class is written in a particular language expressed using the datatype String). As classes, object and data properties can be represented in a hierarchy where properties higher in the hierarchy are more general than their children. Annotation properties are used for providing different types of annotations to the ontology and its constructs. For example, the rdfs:comment annotation property can be used for associating a natural language definition to a class, as illustrated for the *School* class. Another type of annotation property is rdfs:label, which is used to associate a human-readable label description to a class.

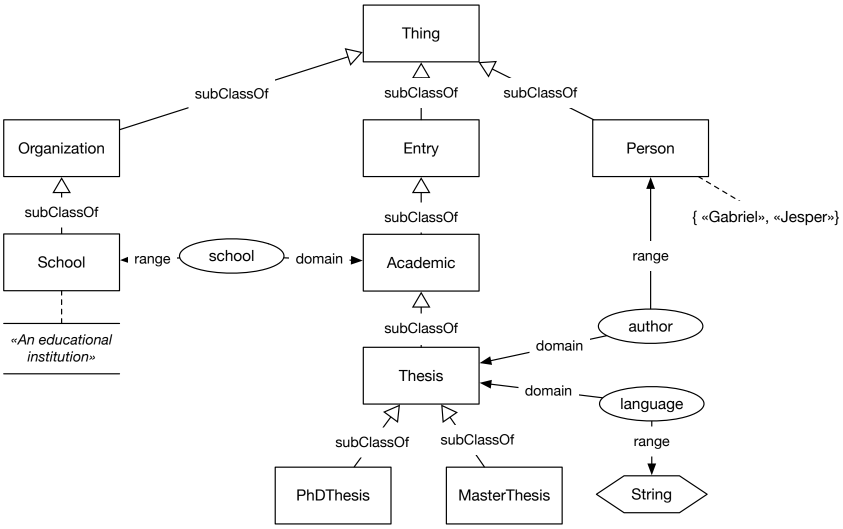


Figure 2.1: An example ontology illustrating various ontology constructs.

Ontologies come in different levels of generality. Guarino [54] suggests the following classification:

- *Top-level ontologies* describe general concepts such as time, space, events, actions, etc. These concepts are domain-independent and can be used for most application purposes. Examples of top-level ontologies are DOLCE [92] and PROTON [135].
- *Domain ontologies* and *task ontologies* describe concepts related to a specific domain or a particular task or activity respectively. These types of ontologies specialise the concepts introduced in the top-level ontologies.
- *Application ontologies* describe concepts that depend both on a particular domain and a particular task, and may correspond to the roles played by domain concepts while performing a specific activity.

In practice, many ontologies represent a blend of the generality levels proposed by Guarino, contributing to mismatches that make the task of aligning ontologies challenging. The next section describes some key concepts related to aligning ontologies, while a description of different types of mismatches that ontology alignment techniques need to deal with are described in Section 2.3.

2.2 Aligning Ontologies

The process of computing alignments between heterogeneous ontologies is often called Semantic Matching, Ontology Matching or Ontology Alignment. Such alignments support the ability to re-use existing ontologies, one fundamental principle in ontology engineering, and more operationally, it supports interoperability among information systems employing the ontologies so that data communicated among them can be interpreted unambiguously. Typically the matching process involves two ontologies to be matched, but in principle, the matching process may involve more than two (this is commonly referred to as multiple matching). The result of a matching process is an alignment artefact, which consists of a set of semantic relations. In this work, the focus is on equivalence and subsumption relations, but other semantic relations exist, such as disjointness and overlap described by e.g. Euzenat [34].

Figure 2.2 extends the ontology example in the previous section by introducing a second ontology which also includes constructs for describing a doctorate thesis (the ontology to the left). The semantic relations discovered between these two ontologies are represented with dotted arrows. Some of them are quite intuitive, such as the equivalence relations between *Thesis* and *Thesis′*, *Organization* and *Organization′*, *Mastersthesis* and *MasterThesis′*, and *Phdthesis* and *PhDThesis′*. The two latter equivalence relations include some syntactic differences that a basic string matching technique would easily resolve. The equivalence relation between *Person* and *HumanAgent′* cannot be identified through string patterns. This relation could be inferred from the fact that both classes have the same individuals (≪Gabriel≫ and ≪Jesper≫) as members. This is typically called *instance-based matching.* Another possibility is to use property patterns to infer this equivalence relation. Both *Person* and *HumanAgent′*are defined as the range of the object property *author* which indicates at least some relatedness between the two classes.

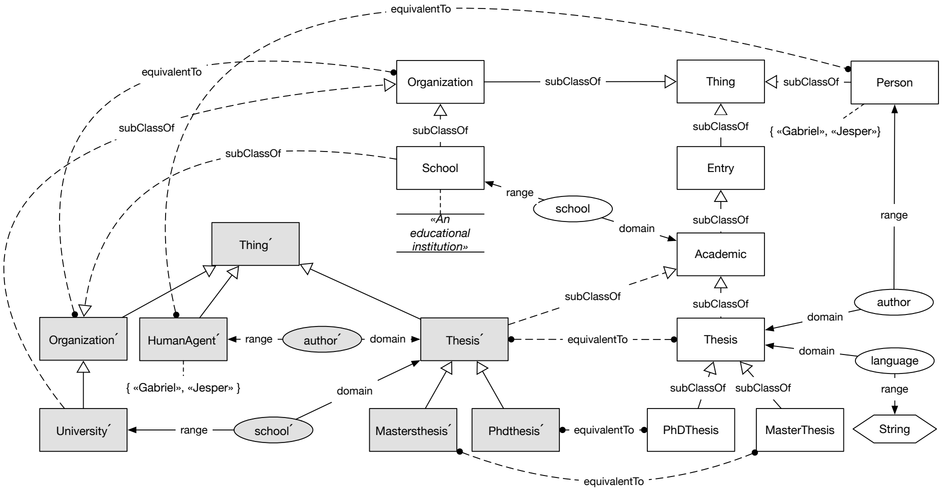
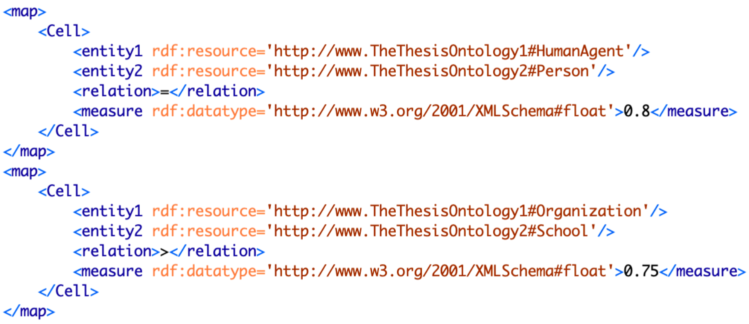


Figure 2.2: An example of equivalence and subsumption relations between concepts in two ontologies

University′ is a subclass of *Organization*. This relation could be inferred from a structural analysis since *University′* is a subclass of *Organization′* and *Organization′* is equivalent to *Organization*. It could also be identified using external sources of knowledge, such as the WordNet lexicon (see Section 2.5.3), that states that University has a more specific meaning than Organization. The same reasoning could be applied to infer that *School* is a subclass of *Organization′* and that *Thesis′* is a subclass of *Academic*. The latter relation could also use the annotation property (rdfs:comment) associated with *School*. Here, the combination of natural language processing (NLP) techniques and a lexicon could be employed to determine that the term *institution* is a more specific term than *organization*, hence *School* should be a subclass of *Organization'*.

Formally, a semantic relation is expressed as a quadruple $< e, e' r, c >$ where e and e' are two aligned entities[5] across ontologies, r represents the type of semantic relation holding between them, and c represents the confidence of the relation between these two entities. Figure 2.3 shows how semantic relations are expressed using the Alignment Format (further described in Section 2.2.2). Each semantic relation is described within the <Cell> element, the entities being matched are described in <entity1> and <entity2>, the relation type (where '=' indicates equivalence and '<' or '>' indicates subsumption) is described in <relation> and the confidence value determined by the matcher is defined in the <measure> tag.



```
<map>
    <Cell>
        <entity1 rdf:resource='http://www.TheThesisOntology1#HumanAgent'/>
        <entity2 rdf:resource='http://www.TheThesisOntology2#Person'/>
        <relation>=</relation>
        <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>0.8</measure>
    </Cell>
</map>
<map>
    <Cell>
        <entity1 rdf:resource='http://www.TheThesisOntology1#Organization'/>
        <entity2 rdf:resource='http://www.TheThesisOntology2#School'/>
        <relation>></relation>
        <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>0.75</measure>
    </Cell>
</map>
```

Figure 2.3: Representation of semantic relations in an alignment artefact.

2.2.1 Sub-processes in Ontology Matching

There are several sub-processes involved in a complete matching process. A typical workflow is illustrated by Rahm [120] as shown in Figure 2.4.

The ontologies to be matched are first parsed so that matchers can compute various types of similarity measures among their concepts. As mentioned earlier, it is common that an ensemble of matchers is used in the *matcher execution*, where each individual matcher identifies semantic relations among concepts based on different ontology characteristics and techniques. The choice of which matchers to include in the matcher ensemble is normally

[5] An entity usually refers to a class (concept), but can also represent properties (object and data) in the case of property matching and individuals in case of instance-based matching.

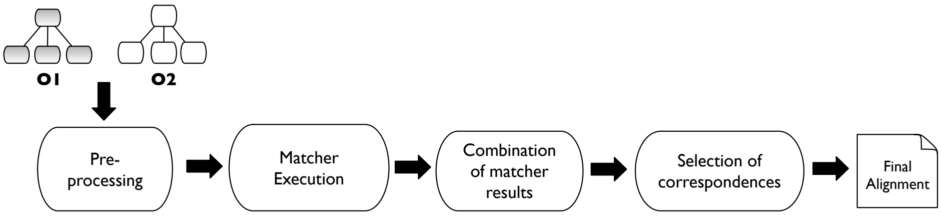


Figure 2.4: A typical workflow of an ontology matching system (adapted from Rahm [120]).

determined manually. The use of several complementary matchers can potentially compensate for the weaknesses of each other [41], but if this ensemble is not correctly composed, it can also decrease the overall quality. Once the appropriate matchers are chosen, they have to be configured and tuned according to the particular characteristics of the ontologies to be matched. The matcher configuration typically includes weight assignment for the different matchers, configuring potential external sources, and deciding a confidence threshold for the resulting alignment. If any of the matchers use external sources, such as the WordNet lexicon, these sources must be selected and configured with appropriate parameters.

Fundamentally, matchers are typically run either in sequence, in parallel or by using some hybrid strategy combining the two, as illustrated in Figure 2.5 (adapted from Rahm [120]). In the case of a sequential strategy, a first matcher computes an alignment which is used as input to a second matcher, and so on. One rationale for using such a strategy is that different matchers have different complexity and run-time performance. Using a fast matcher first, for example a string matcher, to produce an initial alignment, which is then transferred to the more comprehensive (but slower) matchers, will reduce the overall execution time. This strategy is for example used in the YAM++ system [102] where a terminological matcher produces an alignment which is used as input for a structure-based matcher. In the parallel strategy, all matchers are run independently and their proposed relations are transferred into a final alignment. One benefit of this strategy is that the matchers can run in a distributed fashion, possibly on multiple servers or machines. Furthermore, while the sequential strategy puts much responsibility on the first matcher in the sequence, with the risk of losing other correct relations, the parallel strategy distributes this responsibility between the involved matchers. Hybrid approaches combine sequential and parallel strategies. There are alternative workflows for running matcher ensembles, for example as suggested by Trojahn et al. [148] (also described in Section 3.4). In this multi-agent approach the semantic relations computed by one matcher are mediated to the other matchers for verification or counter proposals. This is an iterative approach that runs until there are no more counter proposals from any of the matchers in the ensemble.

After the matchers have been executed and their alignments are produced, there are usually several post-processing steps, in particular, *combining the results from individual matchers* and *selecting the correspondences* (i.e. semantic relations) that should be returned in the final alignment.

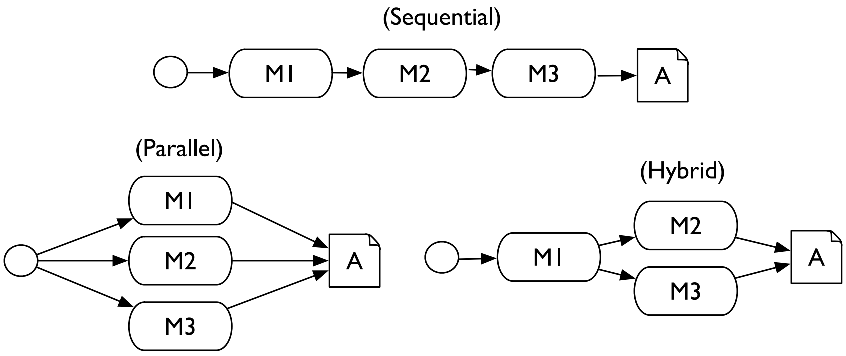


Figure 2.5: Different matcher workflow strategies (Rahm [120]).

2.2.2 The Alignment API

The Alignment API [23] is a Java API for ontology matching. The API offers an infrastructure supporting the development of matching algorithms, generating alignments in a standardised format, manipulating existing alignments, and evaluating alignments, to name a few. The API includes wrappers for interacting with other programming libraries, such as the OWL-API [66], Apache JENA[6] and SKOS[7]. It also includes OntoSIM [8], a library of different similarity techniques.

Tightly coupled with this API, is the Alignment Format[9]. This format has become the de facto standard format for describing an ontology alignment and is used by several matching systems as well as the OAEI evaluation initiative.

[6] https://jena.apache.org/

[7] https://www.w3.org/TR/2008/WD-skos-reference-20080829/skos.html

[8] http://ontosim.gforge.inria.fr/

[9] http://alignapi.gforge.inria.fr/format.html

An extension of the Alignment Format is EDOAL (Expressive and Declarative Ontology Alignment Language)[10]. EDOAL includes a set of constructors and operators used for expressing more precise relations between ontology concepts, such as complex relations (e.g., that a concept in the first ontology is equivalent to the union of two concepts in the second ontology).

EDOAL enables more precise alignments supported by the ability to [35]:

- use algebraic operators to construct entities from other entities. For example, in order to express that a union of entities in one ontology is equivalent to a single entity in the other ontology using the *OR* operator.
- put restrictions on entities in order to narrow their scope. For example, to express that a class in one ontology is equivalent to a class in the other ontology, but only for values defined using a particular object property.
- transform property values. For example, property values using different encodings or units can be aligned using transformations.

2.2.3 Evaluation of Ontology Alignment

Typically, ontology matching systems and techniques are evaluated using the evaluation tracks provided by OAEI[11]. Here, different datasets, both manually constructed and synthetically constructed, are used in different evaluation tracks. The datasets normally consists of a set of ontologies for which a reference alignment (ground truth) holding the correct relations between pairwise ontologies represent the baseline. The alignments computed by the participating matching systems are then compared with these reference alignments.

In general, when evaluating the quality of the alignment the evaluation measures applied are typically precision p, recall r and F-measure fm. These measures are computed with respect to a reference alignment R that holds the true set of correspondences and that is normally manually produced.

Figure 2.6 from the book of Euzenat and Shvaiko [35] along with the below formal descriptions illustrate how these measures are computed.

Precision - Precision p measures the ratio of correct relations in an Alignment A (where correctness is determined by the reference alignment R)

[10] http://alignapi.gforge.inria.fr/edoal.html

[11] http://oaei.ontologymatching.org/

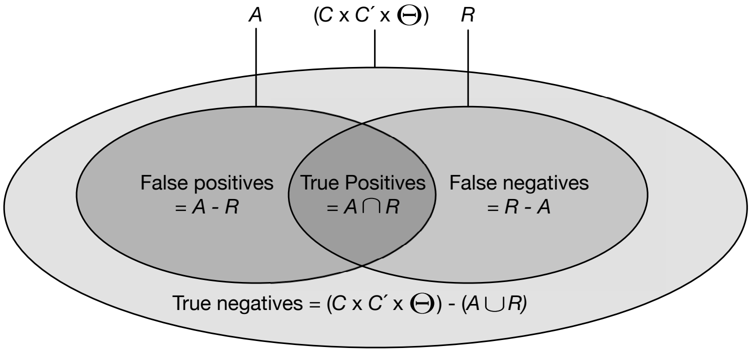


Figure 2.6: Measuring correctness of alignments (Euzenat and Shvaiko [35]).

compared to the total number of relations returned by the matching system.

$$p(A,R) = \frac{|A \quad \cap \quad R|}{|A|} \tag{2.1}$$

Recall - Recall r measures the ratio of correctly found relations in an Alignment A over the total number of correct relations in a reference alignment R.

$$r(A,R) = \frac{|A \quad \cap \quad R|}{|R|} \tag{2.2}$$

F-measure - Given a reference alignment R and a number α between 0 and 1, the F-measure of an alignment A is a function $fm_\alpha : \wedge \times \wedge \rightarrow [0 \quad 1]$ such that

$$fm_\alpha(A,R) = \frac{p(A,R) \times r(A,R)}{(1-\alpha) \times p(A,R) + \alpha \times r(A,R)} \tag{2.3}$$

If α is 1 the F-measure is equal to precision, if it is 0 it is equal to recall, and when it is 0.5 then F-measure represents the harmonic mean of precision and recall [35]. Using an α of 0.5 is common and is also the practice used for all evaluations in this work.

Euzenat [33] proposed a different approach to precision and recall that better complies with the reasoning capabilities offered by ontologies. This approach is called *semantic precision and recall*. Here, semantic relations entailed (by a reasoner) from the merged ontology constructed from a source ontology O_s and a target ontology O_t, as well as a reference alignment RA are considered in the evaluation of an alignment A [163].

Semantic Precision p_{sem} is computed as the number of relations in A that are entailed from the reference alignment RA divided by all relations in alignment A.

$$p_{sem}(A, R) = \frac{|A \quad \bigcap \quad Cn(RA)|}{|A|} \tag{2.4}$$

Semantic Recall r_{sem} is computed as the number of relations entailed from alignment A that are included in the reference alignment RA divided by all relations in a reference alignment RA.

$$r_{sem}(A, R) = \frac{|Cn(A) \quad \bigcap \quad RA|}{|RA|} \tag{2.5}$$

Since such entailments include subsumption inferred from equivalence, semantic precision and recall can be used as measures to compare the performance of matching systems producing only equivalence alignments with systems producing both equivalence and subsumption relations.

The reference alignments used in the different OAEI datasets are constructed according to different modalities. For example, in the Conference track of the OAEI 2019 campaign[12], three different evaluation modalities were applied:

- Crisp reference alignments. Here, the confidence value for all relations in the reference alignment is set to 1.0. Precision, recall and F-measure (F1) as described above is used as-is to evaluate system performance in this modality. There are three different versions of the crisp reference alignments where one is the original (ra1), the second includes entailments and is coherent (ra2), while in the third (rar2) violations of consistency and conservativity are resolved using a combination of tooling and manual assessment.
- Uncertain version of reference alignments. In this version the confidence value of the relations in the reference alignment reflect the degree

[12] http://oaei.ontologymatching.org/2019/conference/eval.html#uncertain-ra

of agreement of a manual assessment of each relation performed by a group of twenty people [17].

- Logical reasoning. Here, violations of consistency and conservativity principles in the evaluated alignments are taken into account.

Based on these different modalities the matching systems participating in the evaluation campaign are evaluated using different metrics and principles. For example, when using the uncertain version of the reference alignments, the involved systems are evaluated based on *discrete* and *continuous* approaches. The discrete approach considers that any relation in the reference alignment having a confidence of $>= 0.5$ to be fully correct, while those with a confidence lower than 0.5 are considered fully incorrect. Furthermore, relations in the reference alignment of the discrete approach have been removed if less than half of the group of people in the manual assessment agreed with them. The matching systems's match is considered correct if the confidence value in the reference alignment is $>=$ to the system's threshold and incorrect otherwise. The continuous approach considers the opinion of the group of people in that it penalises a matching system more if the system does not identify a relation which most people in the group agree on than if it does not identify a relation which has less agreement within the group.

2.3 Ontology Mismatches

Different types of mismatches or heterogeneities make the task of aligning heterogeneous ontologies challenging. One of the assumptions in this work is that semantic matching can learn from theories about what are the properties of different mismatch types and why they occur. If a matching system includes techniques that can identify such mismatches, this might result in better quality alignments. Therefore, it is essential to identify the mismatches that can be solved by automated means and those that require some form of human intervention [141]. Many of the matching algorithms applied are quite naive (e.g., string matching algorithms), and applying principles learned from theories on ontology mismatches, can help filter out false positive relations identified by those naive algorithms, and consequently improve alignment precision.

According to Visser et al. [154] the creation of an ontology involves two sub processes:

1. Conceptualisation - during this process, decisions are made with respect to classes, relations, instances, functions and axioms that are

distinguished in the domain, and the outcome is a conceptualisation that involves these entities. However, the form or appearance of these descriptions is not considered in this process, this is taken care of in the explication process.

2. Explication - during this process, the conceptualisation from the previous process is explicated using some form of ontology language.

These two processes form the conceptual background for much of the literature describing ontology (and schema) mismatches. Within this literature there exists different classifications of such mismatches, at varying levels of detail and with substantial overlap. One classification is from Klein [83], which is illustrated in Figure 2.7. Klein also distinguishes conceptualisation mismatches from explication mismatches, where the former includes mismatches caused by differing *coverage* and *scope*, while the latter refers to differing *terminology*, *modelling style* and *encoding*.

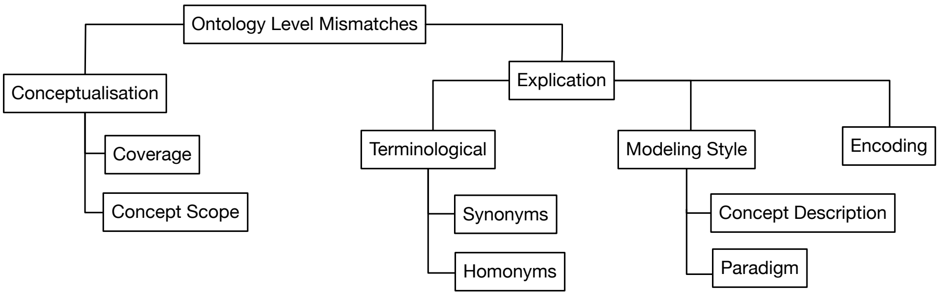


Figure 2.7: Ontology Mismatch Classification (Klein [83])

In the next two sub sections we explain the different types of mismatches included in the classification from Klein. These explanations of mismatches have informed the development of the mismatch detection strategies implemented in this thesis (see Section 5.5).

2.3.1 Conceptualisation Mismatches

Coverage mismatches refer to that two ontologies cover or emphasise different parts of a domain, or that their level of detail differs. *Concept scope* mismatches occur when two classes seem to represent the same concept, but they do not have exactly the same instances, although they may intersect.

Conceptualisation mismatches, called *Conceptual Heterogeneity* by Euzenat and Shvaiko [35], are difficult to identify automatically. Such mismatches occur when there are two (or more) conceptualisations of a domain, and present themselves in how the ontology concepts are distinguished or how they are related.

Visser et al. [154, 153] further decompose conceptualisation mismatches. They distinguish *class* mismatches from *relation* mismatches, where the former are mismatches that relate to classes and subclasses distinguished in the conceptualisation and the latter encompasses mismatches that relate to hierarchical relations (the subsumption hierarchy) and assignment of properties to concepts.

Class mismatches consists of *categorisation* mismatches and *aggregation-level* mismatches. Categorisation mismatches occur when two ontologies include the same class, but decompose them into different subclasses. Aggregation-level mismatches occur when both ontologies include the same concept, but define the concept using classes at different levels of abstraction.

Relation mismatches include *structure* mismatches, *attribute-assignment* mismatches, and *attribute-type* mismatches. Structure mismatches occur when two ontologies distinguish the same set of classes, but differ in how these classes are structured through relations. Attribute-assignment mismatches occur when two ontologies differ in how they relate other classes to the shared concept through object properties. Attribute-type mismatches relate to how properties that are associated with a shared concept use different types.

2.3.2 Explication Mismatches

Explication mismatches relate to how the conceptualisation is specified (explicated).

According to Klein, *Modeling style* mismatches include *paradigm* and *concept description* mismatches. Paradigm mismatches refer to how different paradigms can be used to represent concepts such as time, action, plans, causality, and propositional attitudes. For example, one ontology might use temporal representations based on interval logic, while another might use a representation based on point [13]. Concept description mismatches occur when two similar concepts are modelled differently, for example, that the same intention is modelled through the use of properties in one ontology and by using distinct subclasses for the same target values in the other ontology [13].

Terminology mismatches include *synomym terms* mismatches and *homonym terms* mismatches. *Synonym terms* mismatches occur when identical concepts are represented by different terms (e.g. 'Car' versus 'Automobile').

Homonym terms mismatches occur when the meaning of two identical terms is different (e.g. the term 'Conductor' has a different meaning in the music domain than in the electric engineering domain).

Visser at al. [154, 153] present six different types of mismatches that occur because of different knowledge definitions of the ontologies and their concepts. Here, the definition of knowledge includes three parts: The *term* (T) used to denote a concept, the *definiens* (D) that comprise the body of the definition (e.g. the property statements), and the underlying *concept* (C) itself.

- A *Concept Mismatch* (C) occurs when the definitions have the same terms and definiens, but differ conceptually. Whenever such a mismatch occurs, T is a homonym.
- *A Concept and Definiens Mismatch* (CD) is when the definitions share the same term but have different concepts and definiens. As with the Concept Mismatch, T is a homonym whenever this mismatch occurs.
- A *Definiens Mismatch* (D) occurs if the definitions have the same concept and the same term, but different definiens.
- A *Term Mismatch* (T) is when the definitions share the same concept and the same definiens, but the terms are different. This mismatch implies that the two terms are synonymous.
- *Concept and Term Mismatch* (CT) occurs when the definitions have the same definiens, but differ in their concepts and terms. In this case, the two concepts are most likely different.
- *Terms and Definiens Mismatch* (TD) is when the definitions have the same underlying concept, but the terms and definiens are different. As with the Terms Mismatch this mismatch implies that the terms are synonyms.

Visser at al. [154, 153] stresses that mismatches do not operate in isolation, but that whenever a mismatch occurs between two concepts, this influences the surrounding (sub- and super) classes as well, an effect called the *inheritance of mismatches*. Hence, whether two concepts in fact have a certain semantic relation between them depends in the end on the natural language description of all terms that directly or indirectly contribute to the meaning of these two concepts.

Consider for example the concepts depicted in Figure 2.8[13]. Here, the concept hierarchies are the same, but the lowermost concept 'Cormorant' have differing definiens in that the concept on the left-hand side of the figure refers to a cormorant as a fish-eating, flying and diving bird, while the cormorant to the right-hand side refers to the two copper bird sculptures watching over the Mersey river in Liverpool from the Royal Liver Building.

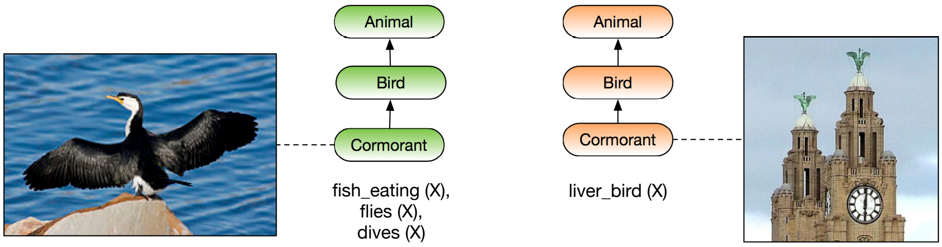


Figure 2.8: Illustration of inheritance of mismatches.

2.4 Semantic Relations

Semantic relations are meaningful associations between two or more concepts, and the underlying meaning of concepts can often be inferred from the semantic relations associated with them [82]. Different semantic relations can inform the identification of each other, hence, it is essential for a matching system to identify a variety of semantic relations. For example, if a matching system identifies a subsumption relation, it is possible to infer equivalence relation(s) for related concepts. This is illustrated in Figure 2.9 where the system identifies that the concept *Car* in ontology O1 is subsumed by the concept *Conveyance* in ontology O2, and that *Automobile* in O2 is subsumed by *TransportMeans* in O1. From these relations, there is a strong likelihood of O1's *TransportMeans* being equivalent to O2's *Conveyance* and that *Car* in O1 is equivalent to *Automobile* in O2.

It is also important to distinguish among different 'non-equivalent' relations. Often, meronymic relations are misinterpreted as subsumption relations [7], yet the semantic interpretation, as well as the usage, of these types of relations are different. While subsumption relations define the notion of specialisation, i.e., that one concept is a specialisation or generalisation of another concept, meronymic (a.k.a. partonomic) relations defines part-whole relations, and are typically expressed by object properties in an ontology. If a system is capable of identifying a part-whole relation (e.g. Component-

[13] The pictures are taken from Wikipedia

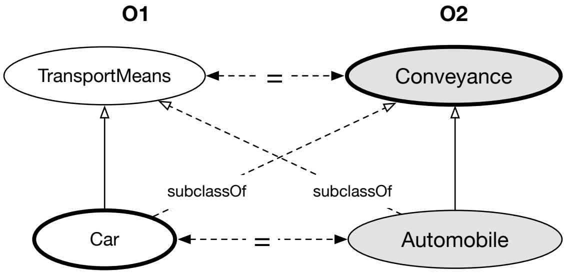


Figure 2.9: Example on how subsumption relations can lead to identification of equivalence relations

Integral part of Object), this can rule out equivalence or subsumption relations involving the same concepts.

Semantic relations play an important role in how knowledge is represented psychologically, linguistically, and computationally [52]. Semantic relations can be distinguished as relations between concepts in the mind (conceptual relations) or relations between words (lexical relations). The semantic relations involved when knowledge is expressed for computational processing are usually called *lexical-semantic relations*. Such relations provide structure to lexicons, thesauri, taxonomies, and ontologies. The main lexical-semantic relations are hyponomy (is-a, broader-narrower, subsumption), meronymy (part-whole), synonymy and antonymy (opposite meaning) [82]. In this work, we focus on the first three, and they will be described in the following. In ontology alignment, the term equivalence is used for representing the binary relation between synonyms, and subsumption is used for expressing hyponymy, so these terms will be used here.

2.4.1 Equivalence Relations

An equivalence relation is a binary relation between two concepts that are considered synonymous, i.e. semantically identical. Most humans, and certainly a string matching algorithm, would say that two concepts $Thesis'$ and *Thesis* could be considered both syntactically and (at least in most cases) semantically equal. That said, there could be situations where two syntactically equal concepts are semantically different. Continuing the above example, the concept *thesis* can, for example, refer to a student thesis (bachelor, master or doctor), a statement in an argument, or a down-beat in a musical play (in contrast to arsis which is the up-beat or unaccented note). This is referred to as *homonymy*, that is, the same word can have multiple unrelated meanings [84]. Related to homonymy is *polysemy*. Here, a word can also have multiple meanings, but in contrast to homonymy, the meanings are related. For example, a *review* as a noun and as a verb (to review something) is a polynom [84]. Hence, it is very much context-dependent whether or not two concepts are semantically equal or not, and determining true equivalent concepts among homonyms and polysemes is challenging.

2.4.2 Subsumption Relations

A *subsumption relation* defines that one concept is a sub-type of another concept, for example that the concept *Car* is a sub-type of the more general concept *TransportMeans*. Such relations are also commonly known as is-a and class inclusion relations [55]. As described in Section 2.1, the fundamental structure of an ontology is a set of concepts organised in a subsumption hierarchy, effectively establishing a hierarchy of concepts that are subsumed by each other, all the way up the root of the ontology (thing). A concept subsumed by another inherits the properties from the subsuming concept (and all subsuming concepts up to the root), therefore we say that a subsumption relation is transitive.

Chaffin et al. (cited in [140]) identified the following kinds of subsumption relations:

1. Natural Object-Kind (Employee is-a Person)
2. Artefact-Kind (Laptop is-a Computer)
3. State-Kind (Single kind-of Marital Status)
4. Activity-Kind (Consulting kind-of Work)

According to Storey [140], the two first kinds of relations are best represented as two concepts connected by a subsumption relation, while 'State-Kind' and 'Activity-Kind' are best represented by making the state or activity a property of an appropriate concept. For example, 'consulting' could be an instance of a class Work which is the range of an object property 'typeOfWork'.

2.4.3 Meronymy Relations

A *meronymy or part-whole relation* models the parts that comprise a whole concept and are often used to relate different classes in an ontology through properties [8]. For example, the object property *isMadeOf* relates instances of the concept *House* to instances of the class *ConstructionMaterial*.

There are different types of part-whole relations, and they have different characteristics. Winston et al. [157] suggest six different types of part-whole relations as presented in Table 2.1.

Table 2.1: Different types of part-whole relations

Relation	Example
Component / Integral Object	handle-cup or punchline-joke
Member / Collection	tree-forest or card-deck
Portion / Mass	slice-pie or grain-salt
Stuff / Object	gin-martini or steel-bike
Feature / Activity	paying-shopping or dating-adolescence
Place / Area	Everglades-Florida or oasis-desert

2.5 Computing Similarity

2.5.1 String Similarity

String similarity methods have a prominent place in most semantic matching systems [105, 15, 139]. These methods are normally fast and are in many situations able to determine that two concepts from different ontologies are equal or similar based on the string representation of the concepts. Ngo et al. [105] classifies string similarity methods into the following categories:

- *Local terminology-based methods* which focus on determining similarity based on individual entities.
- *Global terminology-based methods*, which also consider the context (i.e. neighbouring entities) in which the entities being compared reside or that combine several local methods.

In the following we will focus on Local terminology-based methods and describe some of the methods that are commonly used in semantic matching.

Edit-distance based methods measure the distance between two strings S and T based on counting the number of operations it takes to transform from S to T, where operations include insertion, replacement and deletion of characters in the string [35]. Each operation is assigned a cost such that the distance between the two strings is computed as the sum of the less costly operations that transforms S to T. The *Levenshtein distance* [86], which is one commonly used implementation of edit distance, operates with all operations (insertion, substitution, and deletion) being equal to 1. The *Hamming distance* [61] is a distance metric that counts the number of positions where two strings of equal length differ. If the two strings to be compared are of uneven lengths, one variant is to normalise by the length of the longest string [35]. The *Jaro* [72] distance metric considers the number of matching characters, their sequence, and their length in order to arrive at how similar (or distant) two strings are. The *Winkler* [156] extension consists of adding more weight if the two strings share a common prefix.

A more run-time efficient approach to string matching is the use of substring comparison. The *n-gram* (q-gram) technique converts strings into sets of n-sequences of characters composing the string [15]. For example if n=3 (trigram), the string 'thesis' is transformed to a set {“the”, “hes”, “esi”,“sis”}. Especially for longer string comparisons, such as when comparing two comments associated with ontology concepts, substring similarity techniques such as n-gram can be applied.

ISub [139] is a string matching algorithm targeted for ontology matching. The algorithm applies three functions in order to find the similarity between two entity names C_x and C_y and considers both the commonality and difference between strings when computing a similarity score. The algorithm proceeds as follows:

$$ISubSim(C_x, C_y) = comm(C_x, C_y) - diff(C_x, C_y) + winkler(C_x, C_y) \quad (2.6)$$

The three functions are:

- The commonality function ($comm$) is motivated by the substring metric where the biggest common substring between two strings is computed. This process is further extended by removing the common substring and by searching again for the next biggest substring until no common substring can be found.

- The difference function ($diff$) is based on the length of the unmatched strings resulted from the initial matching step (after the common substrings have been identified). The Diff function is given less importance than the commonality function (weight parameter 0.6 is a good value according to the authors [139]).

- After the commonality and difference between two strings are computed the Winkler algorithm [156] is used for improving the results.

Another family of string-based similarity techniques bases the comparison on tokens. Here, concept names and/or other natural language descriptions of ontology concepts are represented as sets of tokens, also called “bag-of-words”. These tokens can then be used to compute a set-theoretic similarity score or represent a basis for a vector-based similarity technique.

One of the commonly used set-theoretic similarity techniques is Jaccard [70]. This technique is also called intersection over union, meaning that the similarity between two word sets S and T is computed as:

$$JaccardSim(S,T) = \frac{S \cap T}{S \cup T} \tag{2.7}$$

TF-IDF (Term Frequency - Inverse Document Frequency) is a vector-based similarity technique that has been extensively used in information retrieval and commonly also to match ontologies [35]. The TF-IDF weight is a statistical measure that reflects how important a word is to a document in a collection [126]. Here, a word’s importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. The TF-IDF weight consists of two components, TF and IDF. TF measures how frequently a given term occurs in a document, and is often computed as:

$$TF(w) = \frac{Number\,of\;times\;word\;w\;appears\;in\;a\;document\;D}{Total\;number\,of\;words\;in\;the\;document\;D} \tag{2.8}$$

IDF measures how important a word is in the whole collection, and is often computed as:

$$IDF(w) = log\frac{Total\;number\,of\;documents\;D}{Number\,of\;documents\;D\;with\;word\;w\;in\;it} \tag{2.9}$$

The TF-IDF is then computed as the product of these two components.

In the context of semantic matching, the document D can be represented by a “virtual document” that includes contextual information associated to each of the concepts being matched, and w can be represented by words that appear within that context. The context can consist of the label, comments, properties and instances associated with the concepts being matched [104]. Once the virtual documents has been created for each concept in the ontologies to be matched, they can be represented in a vector space model.

Similarity between the ontology concepts can then be computed using for example the cosine measure between the vector representations of the virtual documents. The cosine similarity is a measure that calculates the cosine of the angle between two vector representations. The formula for computing it is:

$$\cos(A,B) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2}\sqrt{\sum_{i=1}^{n}(B_i)^2}} \tag{2.10}$$

We use the following basic example to explain how the cosine similarity is computed. Say we want to measure how similar a concept A in ontology O_S is to concepts B in ontology O_T, where A and B are "virtual documents" represented as vectors holding components according to terms describing the concepts in the respective ontologies. The vectors of the virtual documents representing A and B are as follows: A = [1, 2, 0], and B = [1, 1, 2] and the cosine similarity between A and B is given by:

$$cos(A,B) = \frac{1x1+2x1+0x2}{\sqrt{1^2+2^2+0^2}\sqrt{1^2+1^2+2^2}} = 0.548$$

2.5.2 Structure-based Similarity

Structure-based similarity techniques exploit the structural characteristics of the ontologies to be aligned in order to infer semantic relations between them. Euzenat and Shvaiko [35] classifies similarity methods falling under this category as graph-based, taxonomy-based, and model-based techniques. Graph-based techniques consider the input ontologies as directed, labelled graphs, and the similarity between concepts is determined by their position in the respective graphs and their neighbouring nodes. One example of a graph-based technique, which is often used in semantic matching, is the *Similarity Flooding* algorithm [95], where the ontologies (or other structured models) are represented as labelled graphs. This technique assumes that the similarity of two nodes (representing ontology concepts in the graph) depends on the similarity of adjacent nodes in the graph representation of two ontologies. As illustrated in Figure 2.10, similarity flooding establishes a pairwise connectivity graph (PCG) that takes into account similar labels between nodes in the respective models. In this connectivity graph, nodes are then represented pairwise according to label similarity. From the connectivity graph, an induced propagation graph is created on the basis of how many labels the pair of nodes share. For example, as illustrated in the figure, there is one l2-edge going out from ($a1$, b) in the pairwise connectivity graph. In this case the weight coefficient is set to 1.0 since the similarity of $a1$ to b contributes fully to that of $a2$ and $b2$ (i.e. looking at the two models, there is an l2 relation from $a1$ towards $a2$ and an l2 relation from b towards $b2$). In contrast, there are two l1-edges leaving the pair (a, b) in the connectivity graph, thus, the weight of 1.0 is distributed equally among (a, b), ($a1$, $b1$) and (a, b), ($s2$, $b1$).

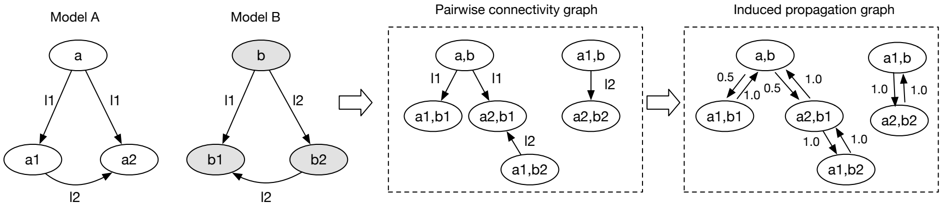


Figure 2.10: Similarity Flooding (adapted from Melnik et al. [95])

From the induced propagation graph, the similarity flooding algorithm is based on an iterative computation of mapping values. The initial mapping values can be calculated by e.g. a string similarity technique (see Section 2.5.1) or with a uniform assignment of 1.0. In every iteration, the mapping values for a map pair (e.g. a and b in the figure) are incremented by the mapping values of its neighbour pair in the propagation graph multiplied by the weight coefficients on the edges going from the neighbour pairs back to the map pair (a, b). The iteration stops when the similarities does not change more than a specified threshold or after a predefined number of steps.

Taxonomy-based techniques consider the subsumption hierarchy (the taxonomy) and assume that concepts that are related via a specialisation relationship are similar and so are their neighbours. An example of a taxonomy-based technique is Wu-Palmer [158]. This technique calculates a similarity score by considering the taxonomical depth of the two concepts to be matched (c_s and c_t), along with the depth of their least common subsumer (lcs):

$$Sim_{wp} = \frac{2 * depth(lcs)}{(depth(c_s) + depth(c_t))} \tag{2.11}$$

Model-based techniques base the similarity computation between two concepts upon model-theoretic semantics assigned to the concepts. In this category we find techniques that apply propositional logic and description logic in order to determine semantic relations between concepts. An example of a system that is based on model-based techniques is LogMap [78]. After having indexed, produced extended class hierarchies, and identified a set of anchor mappings from the indexed entities in the two input ontologies, LogMap encodes the class hierarchies and current relations into propositional logic (based on Horn rules). At this point, a repair process is performed that tries to solve unsatisfiability from the merger of the two input ontologies on the basis of the relations identified so far. This repair process is performed iteratively together with a process for discovering new relations in the context (extended class hierarchies) of the initially identified anchor mappings). The discovery of new relations is performed using the string matching algorithm ISub (described in Section 2.5.1). The output from running LogMap is an alignment that holds a set of relations that will not lead to logical errors when the two input ontologies are merged. Furthermore, LogMap outputs a fragment representing the overlapping between the input ontologies to facilitate manual identification of additional relations that LogMap might have missed.

2.5.3 Lexical Similarity

Most of the better performing matching systems use some form of lexical resource to complement the already mentioned similarity techniques [19]. This is especially important when the task is to identify non-equivalent and asymmetric semantic relations and where the string representation of concepts in many cases cannot serve as an indicator. Many matching systems rely on the *WordNet* lexicon [98] and its database of synsets that help to semantically define and disambiguate concepts. A synset is a grouping of nouns, verbs, adjectives and adverbs where each describes a distinct concept. The synsets, of which there are 117.000 in the current version of WordNet, are interlinked, forming a semantic network that can be queried through a number of available APIs.

Table 2.2 shows the semantic relations that exist in WordNet.

Different approaches to exploiting the lexical database WordNet have been proposed. These approaches can be classified into three main categories [89]:

- Edge-based methods. These methods compute the semantic similarity between two words by measuring the distance (the path linking) of the words and the position of the word in WordNet's taxonomy. Examples of methods belonging to this category are *Wu-Palmer* [158] and *Su* [141].

Table 2.2: Different semantic relations in WordNet

Relation	**Description**	**Example**
Hypernym	is a generalisation of	*motor vehicle* is a hypernym of *car*
Hyponym	is a kind of	*car* is a hyponym of *motor vehicle*
Meronym	is a part of	*lock* is a meronym of *door*
Holonym	contains part	*door* is a holonym of *lock*
Troponym	is a way to	*fly* is a troponym of *travel*
Antonym	opposite of	*stay in place* is an antonym of *travel*
Attribute	attribute of	*fast* is an attribute of *speed*
Entailment	entails	*calling on the phone* entails *dialing*
Cause	cause to	*to hurt* causes *suffer*
Also see	related verb	*to lodge* is related to *reside*
Similar to	similar to	*evil* is similar to *bad*
Participle of	is participle of	*stored* is the participle of *to store*
Pertainym	pertains to	*radial* pertains to *radius*

- Information-based statistics methods. For these methods the basic idea is that the more information two concepts have in common, the more similar they are. Examples of methods based on this category are *Resnik* [121], *Lin* [88].

- Hybrid methods. These are methods that combine principles from the above categories. Examples are *Jiang-Conrath* [75], *Rodriguez* [124] and *Petrakis* [111].

In an experimental evaluation that measured different WordNet-related similarity methods, among them Lin and Resnik mentioned above, Jiang-Conrath was found to outperform the other methods [12]. Being a hybrid method, Jiang-Conrath propose a model that uses the information content as a decision factor in a derived edge-based approach. The information content of a concept derives from the assumption that the more abstract a concept is, the less information it holds [121], or in other words, a more specific concept (such as Festival) has more information content than a general concept (such as Event). The information content is often computed based on how many times a concept is found in a text corpus. In WordNet, the frequency of a concept is incremented each time a particular concept is present, and the same are the ancestors of that concept. This approach is used since each occurrence of a more specific concept also implies that the more general ancestor concept occurs [110].

The Resnik definition of information content is computed as:

$$IC(c) = -ln(\frac{freq(c)}{freq(root)}) \tag{2.12}$$

where *freq(c)* and *freq(root)* represent how many times concept c and the root occurs in a corpus [159].

Another variant of information content is the so-called *intrinsic information content* [130] which does not rely on usage statistics of concepts in a corpus. The intrinsic information content of a concept c is computed as follows:

$$IC(c) = 1 - \frac{log(Sub(c) + 1}{log(|C|)} \tag{2.13}$$

where $Sub(c)$ indicates the number of subclasses of the concept c and $|C|$ represents the total number of concepts in the ontology.

The Jiang-Conrath algorithm [75] is based on finding the information content of both concepts to be matched as well as the information content of the least common subsumer (LCS), that is, the lowest node in the hierarchy that is the parent node of both concepts. The distance function between two concepts c_1 and c_2 is calculated as follows:

$$distance_{jc}(c_1, c_2) = IC(c_1) + IC(c_2) - 2 \cdot IC(lcs(c_1, c_2)) \tag{2.14}$$

where $lcs(c_1, c_2)$ is a function for finding the least common subsumer of both concepts c_1 and c_2.

The similarity between two concepts c_1 and c_2 can then be calculated as follows:

$$sim_{jc}(c_1, c_2) = \frac{1}{distance_{jc}(c_1, c_2)} \tag{2.15}$$

2.5.4 Machine Learning Techniques

Machine learning methods are often classified under *supervised* or *unsupervised* learning. Supervised learning implies that the computer is presented with training data comprising examples of input vectors along with corresponding target vectors [10]. When the task is to distribute the input data into a finite number of discrete categories (such as match | no match), as is often the case in ontology matching, this is called a classification problem. In other words, for two concepts C_s and C_t the task is to learn the mapping between the two concepts given a set of correct values provided by a supervisor (a trained classifier).

Several matching systems use machine learning as part of the matching process and both supervised and unsupervised strategies are applied. This section will present a few examples on how such learning techniques can be applied to aligning ontologies.

In Ngo et al. [102] the authors explain in detail the approach using a supervised *Decision Tree* learning model. This model aims to learn which similarity technique combinations and similarity thresholds that should imply a match / non-match decision for an arbitrary relation pair. Resulting from a training phase using ontologies along with their reference alignment, a decision tree consists of the involved similarity techniques as non-leaf nodes and their similarity scores generated from the training as leaf-nodes. First, similarity scores for each pair of (untrained) relations in the two ontologies to be mapped are computed by the involved similarity techniques. Then, each relation pair starts from the top of the decision tree, and depending on how the score computed in the previous step compares to the score from the training phase, different paths through the nodes in the decision tree are taken, resulting in that the relation pair is either a match (score of 1.0) or non-match (score of 0.0). In the example presented in Figure 2.11, which is adapted from Ngo et al. [102], the relation $Thesis - PhDThesis$ is considered a match from passing through these nodes in the tree: [01-03-05-06-08-10].

In unsupervised learning the input vectors have no corresponding target vectors, hence the challenge is rather to try to identify commonalities in the data without supervision, for example using a clustering algorithm [3].

Hu et al. [69] uses unsupervised clustering to align ontologies using the Falcon-AO matching system. The goal of their approach is to match large-scale ontologies containing thousands of concepts and the clustering technique is used for partitioning the ontologies to be matched into smaller blocks to reduce the number of pairwise comparisons needed for producing an alignment. This approach is based on three overall processes:

1. Partition ontologies into clusters using an agglomerative (bottom-up)

Similarity scores			
Relation	SimTech#1	SimTech#2	SimTech#3
Thesis - PhdThesis	0.3	0.8	0.9

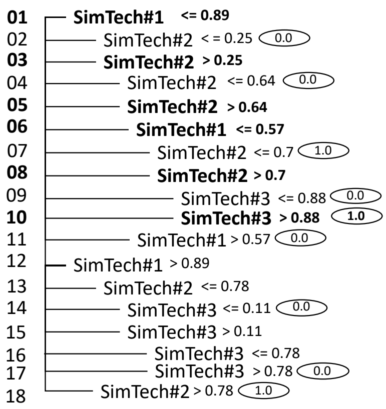


Figure 2.11: Decision Tree Example (adapted from Ngo et al. [102])

clustering algorithm based on structural proximity parameters (see Section 3.1.2),

2. Construct blocks from these clusters based on an analysis of how different entities take part in RDF sentences, and match the blocks based on already computed mappings (so-called anchors) by a fast string matching algorithm (ISub as presented in Section 2.5)

3. Compute alignments between the blocks using a vector-space approach based on TF/IDF (see Section 3.1.1) and structural matching.

The final example in this machine learning section is *word embedding*. Word embedding is an approach that is often considered semi-supervised due to how it aims to learn associated semantics to words based on their interaction with other words in a (large) pre-defined corpus. Here, words are represented with continuous vectors of a fixed *embedding size d* from some vector space in $\mathbb{R}^d$. These vectors are called *word vectors* or *words embeddings*, since each word is "embedded" in a lower-dimensional continuous vector space. This vector space represent a latent feature space and, within this space, each vector represents the lexical semantics of the corresponding word.

Different approaches have been proposed to efficiently learn the latent features space in which such vectors are defined. Mikolov et al. [96] proposed two different models to learn such representations for a task of semantic similarity, Skip-Gram and Continuous bag-of-words (CBOW). Both of them are different realisations of the same fundamental idea, that is, if you have two words that have similar neighbours, then it is likely that these two words are similar too. The two models are illustrated in Figure 2.12 [97].

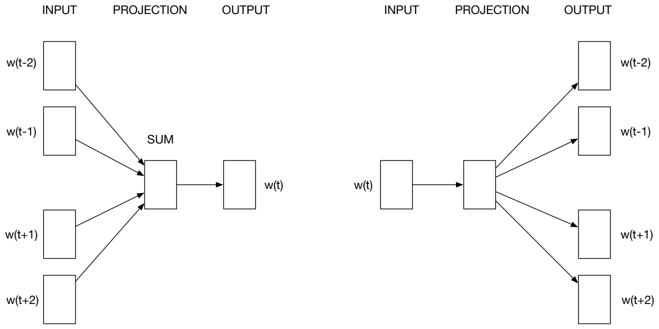


Figure 2.12: The Continuous bag-of-words (CBOW) and Skip-Gram models (Mikolov et al. [96])

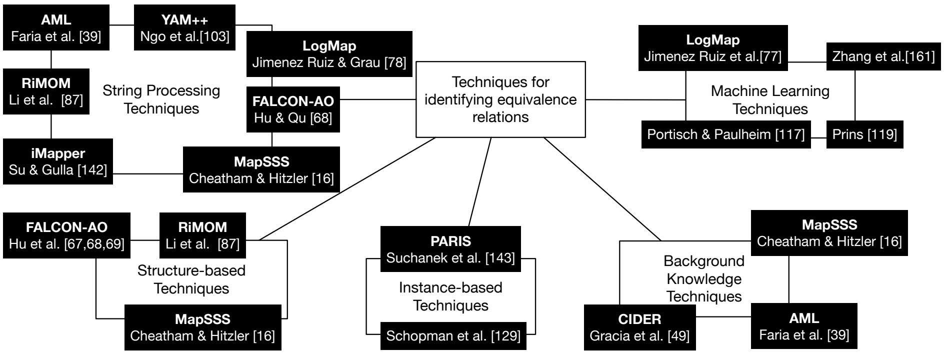
The first model, the Skip-Gram model, has been used to learn distributed representations of sentences through the composition of atomic word vectors. The main intuition behind this model is that, given a word w at the k-th position within a sentence, a neural network is trained to predict the most probable surrounding context. The second model, Continuous bag-of-words, differs from the Skip-Gram model in that it tries to predict the current word from a window of surrounding words.

In the context of ontology alignment, Section 3.1.5 describes some approaches that use word embeddings to infer semantic relations between ontology concepts. Word embedding is also used by two matchers developed in this thesis and an explanation of their approach is described in Sections 5.2.1 and 5.2.2.

3 Related Work

3.1 Techniques for Identifying Equivalence Relations

Over the last two decades a large number of systems and techniques have been developed for equivalence matching and comprehensive surveys of such systems and techniques are well covered in the “Ontology Matching” book by Euzenat and Shvaiko [35] and literature reviews by Otero-Cordeira et al. [109] and Anam et al. [5]. In this section we highlight some of the basic techniques that are used when computing equivalence relations. Figure 3.1 shows a map of these techniques along with a reference to examples of systems and papers describing different approaches.



AML Faria et al. [39]
YAM++ Ngo et al.[103]
LogMap Jimenez Ruiz & Grau [78]
RiMOM Li et al. [87]
String Processing Techniques
FALCON-AO Hu & Qu [68]
iMapper Su & Gulla [142]
MapSSS Cheatham & Hitzler [16]
Techniques for identifying equivalence relations
LogMap Jimenez Ruiz et al.[77]
Zhang et al.[161]
Machine Learning Techniques
Portisch & Paulheim [117]
Prins [119]
FALCON-AO Hu et al. [67,68,69]
RiMOM Li et al. [87]
Structure-based Techniques
MapSSS Cheatham & Hitzler [16]
PARIS Suchanek et al. [143]
Instance-based Techniques
Schopman et al. [129]
MapSSS Cheatham & Hitzler [16]
Background Knowledge Techniques
CIDER Gracia et al. [49]
AML Faria et al. [39]

Figure 3.1: Overview of techniques for detecting equivalence relations.

.

3.1.1 String Processing Techniques

String processing techniques are almost always represented in a matching system [15]. This section describes some of the most common techniques, and in the final part of the section we present a general overview of their performance in one of the OAEI datasets.

Ngo et al. [103] employs a series of string processing techniques during the terminological matching of ontology concepts in the YAM++ matching system. This matching system distinguishes four different categories of techniques (see Section 2.5.1 for an overview of these techniques): edit-based techniques, token-based techniques, hybrid techniques that combine edit-based and token-based techniques, and finally vector-space techniques that use TF-IDF to compute a similarity based on the context associated with the concepts being matched. The context in this case is represented by three different context profiles: (1) the individual profile that concatenates a concept's name, label and comments, (2) the semantic profile that combines the individual profile of the concepts being matched with their respective neighbors' individual profiles, and finally, (3) the external profile that combines the textual description of instances that are members of the concepts being matched. These profiles combined represent each ontology concept as a "virtual document" that can be represented in a vector space. A weight that reflect the importance of each word in the virtual document is computed using TF-IDF.

The AgreementMakerLight (AML) system developed by Faria et al. [39] also uses a series of string processing techniques to match ontologies. One string-based matcher used by AML is the Word Matcher, which measures the similarity between two class names through a weighted Jaccard index (see Section 2.5.1) between the words present in the class names. An example of a more complex string matcher used by AML is the Parametric String Matcher which uses a variety of similarity metrics such as Jaro-Winkler, Levenshtein and Q-gram[1].

LogMap (Jimenez-Ruiz and Grau [78]) computes equivalence relations based on the iteration of two core processes: map discovery and mapping repair (described more in detail in Section 2.5.2). From an initial set of anchor mappings represented in inverted indices, LogMap discovers new mappings by measuring similarity among entities that are semantically related to entities represented in the anchor mappings using the ISub [139] string matching algorithm.

[1] https://github.com/AgreementMakerLight

The RiMOM system (Li et al. [87]) includes two string-based matching techniques. The first is a strategy that combines the use of Edit-distance with a technique that exploits the WordNet lexicon [145]. The second technique is based on creating a vector representation of the context (i.e. the metadata, properties, sub-classes and instances) associated with ontology concepts which is considered a document D (see Section 2.5.1 for a general explanation of the vector-space approach).

A similar vector-based approach is used in the iMapper system. Here, Su and Gulla [142] used TF-IDF to define feature vectors for representing ontology concepts in a vector space. The ontology concepts are first enriched by linking them to relevant text documents using a linguistic classifier. Then in the next step, TF-IDF is used for representing vectors of words in the documents assigned to each ontology concept. Similarity between pairwise ontology concepts is calculated using the cosine similarity of their feature vectors.

Hu and Qu [68] followed a similar approach in the V-doc technique that was used in the Falcon-AO system. Similar to the abovementioned approach, a virtual document is created by accumulating the context to a concept. An adaptation is that the RDF structure associated with the concepts to be matched is exploited to obtain information from neighbouring entities.

MapSSS (Cheatham and Hitzler) [16]) dynamically chooses a string matcher based on defined heuristics derived from an experiment of different string matching techniques and whether precision or recall should be prioritised. For example, if precision is the priority, and the labels of the concepts to be matched are represented by less than two words, Jaro-Winkler is used, however if the priority is recall and the concept's labels consist of more than two words and they contain synonyms, Soft TF-IDF is chosen.

It is difficult to state accurate performance measures for the techniques described above, since they may be subject to modification, they are often combined with other basic techniques, and their performance is very much dependent on the datasets they are run on. Nevertheless, having *some* sense of their performance is of interest. Cheatham and Hitzler [15] did a comparative evaluation of some of the most common string processing techniques, including some of those described above. Figure 3.2, which is adapted from a report [14] containing more detailed evaluation results than in [15], shows their F-measure (F1) scores when identifying equivalence relations among 16 ontologies in the OAEI 2012 Conference dataset.

This illustration also shows the effect of using pre-processing strategies

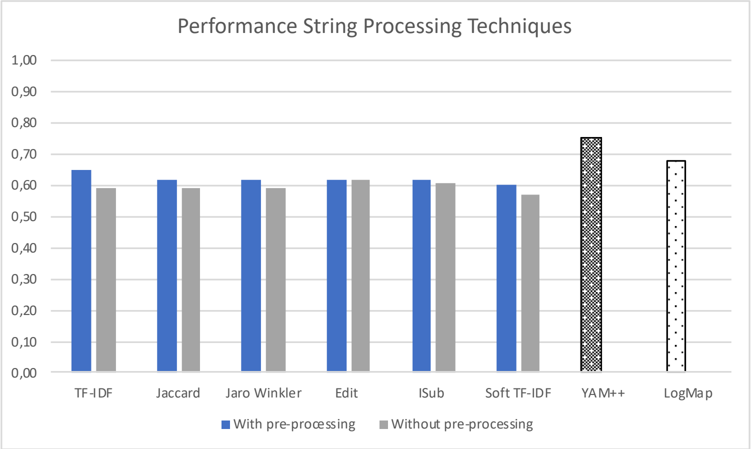


Figure 3.2: Performance of String Processing Techniques (adapted from Cheatham and Hitzler [14]).

before the matchers are run. The scores in blue are obtained using pre-processing (e.g. tokenisation and normalisation), while the scores in grey are without any pre-processing involved. As the figure shows, these techniques, when run in isolation, obtain an F-measure of around 0.6, and for most of them, with some improvement when pre-processing techniques being applied. In order to see how these techniques compare against more complete matching approaches, we see that the aforementioned YAM++ and LogMap systems achieve F-measures of 0.75 and 0.68 respectively on the same dataset.

3.1.2 Structure-based Techniques

The RiMOM system [87] uses a similarity propagation technique similar to the Similarity Flooding approach described in Section 2.5.2. The system includes three different similarity propagation strategies: concept - to - concept propagation, property - to - property propagation and concept - to - property propagation. RiMOM was evaluated using the benchmark dataset of OAEI 2006, where one of the evaluation goals was to see the effect of the similarity propagation. The evaluation results showed that when using similarity propagation the F-measure increased from 0.86 to 0.92.

In the Falcon-AO matching system, Hu et al. [69] use a technique called *structural proximity* to detect equivalence based on the structural properties of two concepts to be matched. The result from the structural proximity computation is then used as input to a clustering-based partitioning process that decomposes the input ontologies into different modules to be matched. Such a decomposition is efficient when matching large ontologies. The structural proximity between ontology concepts is based on how closely related these concepts are in the subsumption hierarchy. The structural proximity between classes c_i and c_j is computed as follows:

$$structProx(c_i, c_j) = \frac{2 \cdot depth(c_{ij})}{depth(c_i) + depth(c_j)} \tag{3.1}$$

where c_{ij} is the (least) common superclass of c_i and c_j, and $depth(c_i)$ and $depth(c_j)$ is the depth of these two concepts in the asserted ontology hierarchy. With this formula the intuition is that the deeper in the hierarchy the common superclass is, the semantically closer c_i and c_j are, and, the structural proximity of the two classes is stronger the deeper in the hierarchy they reside. This is illustrated by the example in Figure 3.3 (where the numbers in parenthesis signify depth). Here, the structural proximity between 'Man' and 'Woman' is 0.67 $(2 \cdot 2/3 + 3)$, whereas the structural proximity between 'Human' and 'Animal' is 0.5 $(2 \cdot 1/2 + 2)$.

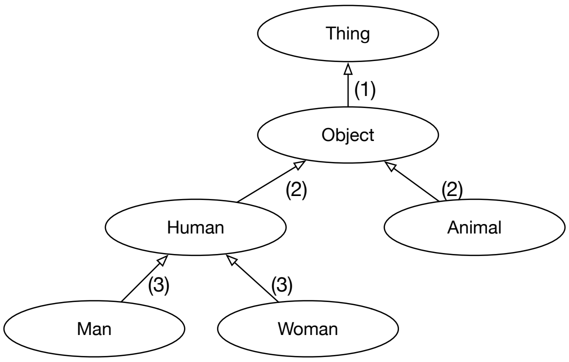


Figure 3.3: Example illustrating the structural proximity approach.

An alternative approach to the structural proximity technique described above is to use the average distance of the common superclasses of c_i and c_j instead of the single least common subsumer as described in Equation 3.1. This latter approach is used by the Graph Equivalence Matcher (GEM) implemented in this thesis (see Section 5.2.5).

Hu et al. [68, 67] also implemented another structure-based matching technique called GMO in the Falcon-AO matching system. This is an iterative structural technique that uses RDF bipartite graphs to represent ontologies. GMO infers equivalence relations on the basis of already produced alignments provided by other matching techniques. The approach then incrementally generates additional alignments by computing structural similarities between domain entities and statements (triples) in ontologies by recursively propagating similarities in the bipartite graphs. GMO was evaluated using the benchmark dataset in OAEI 2005. Without any input mappings as a basis, GMO obtained an average precision and recall of 0.62 and 0.59 respectively, however as the percentage of input mappings increase, as does the performance of GMO. One remark made by the authors is that GMO does not perform well on its own if the two ontologies to be matched have different structural characteristics. Then the approach relies on additional input mappings from other matchers.

The structure-based technique applied in the MapSSS system [16] is based on the direct neighborhood of the concepts to be matched. If all entities in the direct neighborhood of two classes are mapped to one another, then those two classes are also mapped. This is done repeatedly until no new mappings are created.

3.1.3 Instance-based Techniques

Instance-based matching encompasses methods where instance data are used as a means to identify alignments between ontology concepts. In the Paris system, Suchanek et al. [143] used a combination of instances, relations and classes and the *functionality of properties* to determine whether two instances (and consequently the classes they are members of) are related. Functionality in this respect means how indicative the properties are based on the number of incoming links applying them. For instance, a property such as *hasPassportNumber* has high indicative strength as an instance of a particular person has only one passport number. On the other hand, *bornInCity* has low indicative strength as many person instances might be born in a particular city. The similarity measure used in this study was a very simple Boolean string comparison; either two strings are identical or not. The Paris system was evaluated using the *Person-Restaurants (PR)* benchmark of the instance matching track in OAEI 2010. This benchmark consists of three pairs of ontologies populated with instances. Paris obtained an F-measure of 1.0 for the instance-based (class) matching on the restaurants dataset. However, the focus of this track is to identify equivalence among instances (so-called instance matching), and not classes, hence there is no basis for comparing against the other systems competing in this track since their reported scores are based on how well they identify equivalent instances. In another dataset consisting of the two ontologies Yago and DBPedia, Paris obtained a precision of 0.94 (where Yago concepts subsume DBPedia concepts) and 0.84 (where DBPedia concepts subsume Yago concepts). Since there was no reference alignment for this dataset, recall could not be measured.

Often, a set-theoretic similarity measure among instances of the two concepts to be matched is applied in instance-based ontology matching. For example, in Schopman et al. [129] an enrichment strategy combined with the set-theoretic similarity measure Jaccard (see Section 2.5.1) was used in the instance-based ontology matching. First, they used the Lucene[2] search engine to match instances from two different ontologies as follows: For each instance I_s in the source ontology O_s, the most similar instance I_t of the target ontology O_t is automatically classified to the concept in O_s having I_s as a member. Next, Jaccard was used to measure the overlap between the instances of concepts. An evaluation of the precision of the approach was conducted on a dataset involving two thesauri. The first one, the GTAA thesaurus, is used for annotating multimedia materials, while the second, the Brinkman thesaurus, is used to annotate books. 1000 relations produced by the proposed instance-based matching approach were evaluated manually. The results showed that the proposed approach obtained a precision of around 0.72 and an additional insight developed from the evaluation was that even if two concepts are considered lexically equivalent they often do not have similar extensional semantics.

3.1.4 Background Knowledge Techniques

Different forms of background knowledge, i.e. external lexical resources or external ontologies, are often used to support semantic matching. Indeed, strategies for automating such identification can seldom rely on terminological matching techniques alone, but needs to include external sources like the WordNet lexicon or other external ontologies. For example, a string matcher would in most cases consider the term ’stable’ to be equivalent with the term ’table’ (false positive), and not be able to infer that ’chair’ has a semantic relation with ’seat’ (false negative). Both these examples could be managed with the use of appropriate background knowledge from external sources.

The MapSSS system [16] uses a background knowledge strategy that involves google searches to identify equivalence relations. Based on a google search for the label of a source concept, the technique inspects the snippets on the first page of results for the label of the target concept. If such a label is found, a candidate relation is suggested. Then it searches google for the

[2] https://lucene.apache.org/

label of the target concept, and if the label of the source concept is located in the snippets on the first page of this search, the relation is confirmed and added to the alignment. Due to query limits enforced by the Google API, this approach does not work well when matching large ontologies.

AgreementMakerLight (AML) [39] uses WordNet, Uberon (an ontology for anatomy) and Doid (an ontology for human diseases) as background knowledge. A fundamental data structure in AML is the *Lexicon*, which stores lexical information such as local names, labels, and synonyms associated with the ontology concepts. The mentioned background knowledge sources are used when constructing the lexicon, which is used by several of the AML matchers. In addition, the lexicon includes a provenance weight based on for example what type of synonymy relation synonyms associated with the concept have. This provenance weight is used by any matcher that uses the lexicon data structure in AML.

Cider [49] consults both WordNet and external ontologies in order to represent the context of ontology concepts. This context includes synonyms, hypernyms, hyponyms, properties, domains, as well as other information inferred transitively using a lightweight reasoner. From this context, string matching- and structural techniques are used for computing a set of similarities which are later combined using an approach involving an artificial neural network (ANN).

3.1.5 Machine Learning Techniques

Over the years a number of different machine learning approaches to semantic matching have been proposed [101]. A more recent introduction in this area is the use of learned vector representations called *word embeddings* (see Section 2.5.4 for an explanation of this concept).

Jiménez-Ruiz et al. [77] experimented with the use of embeddings as supporting means of decomposing the matching of large ontologies into a set of matching sub-tasks represented by *locality-based ontology modules* [51]. Following from the ontology representation strategy used by the LogMap matching system (see e.g. [78]), all entities from two input ontologies are lexically encoded into an *inverted index*. In order to make the matching task less challenging (e.g. with respect to memory constraints and run-time performance), the entities in the inverted index are further divided into smaller clusters (i.e. matching sub-tasks). In one of their clustering strategies they used neural embeddings as an attempt to create more accurate clusters, that is, clusters with less overlap, than an alternative naive approach. The comparative evaluation of these two candidate strategies showed that the resulting clusters were slightly better in terms of both cluster size and cluster coverage as well as alignment quality (F-measure) when using the neural embedding approach.

Zhang et al. [161] describe a hybrid approach that combines the use of edit distance and word embedding. For each pair of ontology entities edit distance and cosine distance are computed in parallel. Edit distance is computed taking into account names, labels and comments, and cosine distance between the corresponding embedding vectors is used for the word embedding approach. For each pair of entities the maximum similarity measure for these two methods then determines which pair of entities belong to a relation in the computed alignment. The training of the embeddings is not explained in detail, but they have considered only words that occur less than 5 times in the full Wikipedia corpus, and 50-dimensional embedding vectors were produced for each word. This approach by Zhang et al. is evaluated using the OAEI 2013 Conference dataset, where it is compared with other basic techniques exploiting WordNet (Wu Palmer, Lin and Jiang-Conrath), a latent semantic analysis (LSA) technique, as well as using only word embeddings without the support of edit distance. The evaluation results are shown in Table 3.1, adapted from [161]. As the table indicates, the approaches using word embeddings, both when combined with edit distance and without, obtains good precision, and the hybrid approach performs better than when only using the word embeddings. When compared with the best performing matching system in OAEI 2013, YAM++, the word embedding approach achieves a higher precision. However, the recall is much lower, resulting in an F-measure well below that of YAM++.

Table 3.1: Evaluation of using a hybrid approach of Word Embeddings and edit distance

Methods	Precision	Recall	F-measure
WordNet (Wu Palmer)	0.860	0.484	0.618
WordNet (Lin)	0.786	0.469	0.587
WordNet (Jiang-Conrath)	0.770	0.462	0.578
LSA	0.876	0.462	0.605
Word Embeddings	0.872	0.469	0.610
Word Embeddings and Edit	**0.875**	**0.482**	**0.622**
YAM++	0.80	0.69	0.74

Prins [119] also used word embeddings for identifying equivalence relations between ontology concepts. In his work he used an extension of the Skip-gram model (see Section 2.5.4) that aims to differentiate between different meanings of the same word (multi-sense embeddings). However, according to the evaluation results this approach did not improve the results compared to traditional Word2Vec techniques. The corpus used in his work was generated from a random walk algorithm that from a graph representation of the input ontologies retrieves labels from nodes and edges from the input ontologies whilst iterating them. In the experimental evaluation, datasets from OAEI 2015 including anatomical and medical ontologies are used and AgreementMakerLight is used as a baseline matcher. The evaluation results revealed that even if this approach can obtain higher F-measure scores in the anatomy dataset than some of the basic matchers of AML when run in isolation, it is not able to compete with the full ensemble of AML matchers.

Portisch and Paulheim [117] experimented with creating word embeddings from LOD triples hosted in the WebIsA[3] knowledge base and using these as background knowledge when computing equivalence relations with their *ALOD2Vec Matcher*. The WebIsA knowledge base consists of hyponymy triples (e.g. *aircraft skos:broader transportation equipment*) extracted from Common Crawl[4], a freely available corpus crawled from the Web. In order to compute a similarity between two ontology concepts, the two concepts are first mapped to concepts in the WebIsA knowledge base using string matching. Then, if mapped concepts are identified, RDF2Vec [122] is applied to generate embedding vectors for each concept. Finally, the Cosine measure is used to determine similarity between the vectors to infer equivalence relations. Evaluation results show that exploiting the WebIsA data is challenging due to noisy, subjective and inconsistent facts in the knowledge base.

3.2 Techniques for Identifying Subsumption Relations

Figure 3.4 shows a classification of techniques often used for the automatic identification of subsumption relations. These techniques are described in the sub-sections that follow.

[3] http://webisa.webdatacommons.org/
[4] http://commoncrawl.org/

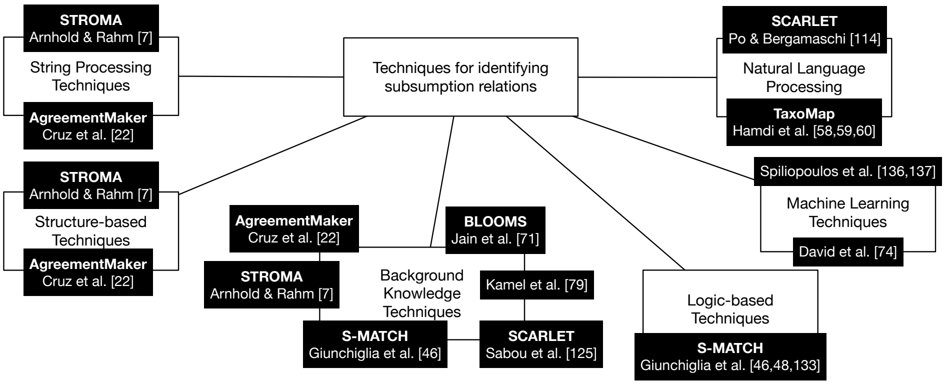


Figure 3.4: Overview of techniques for detecting subsumption relations.

3.2.1 String Processing Techniques

Exploiting compound patterns in concept names is a well-known strategy for inferring subsumption relations, and this technique is used by both Cruz [22] in the AgreementMaker system and by Arnold and Rahm [7] in their system called STROMA. A compound is a word W that consists of a compound head W_H that carries the basic meaning of W and a compound modifier W_M that specifies W_H. For example, the word *ElectronicBook* is a compound where *Book* is the compound head, and *Electronic* is the modifier. The general pattern is that if a source concept C_s is a compound and its compound head equals the target concept C_t, then C_s represents a specialisation of C_t and is thus subsumed by C_t.

The STROMA system [7] use the compounding strategy in most of its matchers, due to its cross-language versatility and effectiveness to identify subsumption relations. However, they note that the recall obtained by this strategy can be low because of the different ways an is-a relation can be expressed. This is also stated by Cruz et al. [22] whose evaluation including eight Linked Open Data (LOD) ontologies revealed that of the techniques they applied for detecting subsumption relations, the compound noun analysis (CNA) technique obtained the lowest recall. The reason for this is according to Cruz et al. that the extracted compound heads cannot usually be matched with the target concepts. The evaluation scores for the compound noun analysis technique are shown in Figure 3.5 (illustration taken from Cruz et al. [22]).

In the Compound Matcher developed in this thesis (see Section 5.3.1), we try

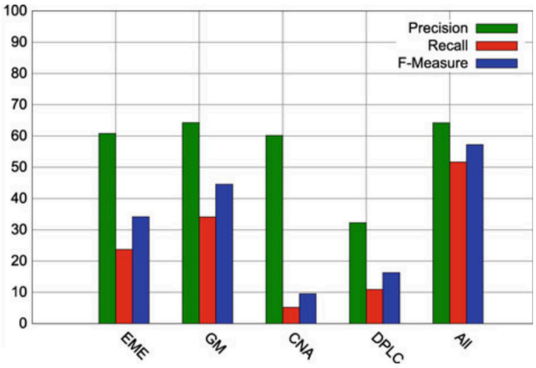


Figure 3.5: Overview of techniques for detecting subsumption relations.

to circumvent the limitations described above by also considering synonyms to the head of a compound word.

The *Itemization Strategy* described in Arnold and Rahm [7] combines string processing and the use of background knowledge. This technique is used when a concept name is an itemization, that is, a list of items where an item is a word or phrase containing commas, slashes or the words “and” and “or”, such as *'books, ebooks, movies, films, cds'* and *'novels and cds'*. This strategy considers the itemized concept names as separate sets of items, and first removes synonym or hyponym concepts within each set (e.g. removing *films* in the above example since it is synonymous with *movies*, and *ebooks* since it is a hyponym of *books*). Then synonym pairs between the two item sets are removed (in the example *cds* are removed). The final processing step is removing hyponyms if there exists a corresponding hypernym in the other set, so *novel* is removed since this is a hyponym of *books*. The relation type is finally determined based on the contents of the remaining sets as follows:

- If both sets are empty, then there is an equivalence relation between the two concepts.
- If one set is empty while the other is not, the empty set is subsumed by the non-empty set, hence a subsumption relation.
- If both sets are non-empty, the relation type is undecided.

According to Arnold and Rahm [7], the itemization strategy can identify the relation type between complex concepts, where other strategies fail.

However, in most ontologies, concept names are normally not expressed as a list of items, and such a strategy is likely not very relevant when matching formal ontologies.

3.2.2 Structure-based Techniques

The *Structure Strategy* by Arnold and Rahm [7] determines a subsumption relation based on the subsumption hierarchy of the ontologies. If two concepts S and T are to be mapped, this strategy infers that if T is equal to the superclass of S, then T subsumes S. This approach is illustrated in Figure 3.6. Here, the concept *Convertible* in ontology *O*1 is to be matched with concept *Car* in ontology *O*2. Since the superclass of *Convertible*, *Car* is equivalent with *Car* in ontology *O*2, it is inferred that *Convertible* is subsumed by (is a subclass of) *Car* in *O*2.

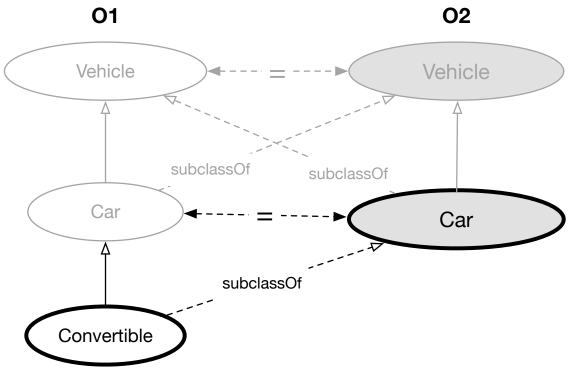


Figure 3.6: Structure Strategy for inferring subsumption relations.

The same approach is used by the Equivalence Mappings Extension (EME) matcher proposed by Cruz et al. [22]. Here, the focus is on mapping Linked Open Data (LOD) ontologies. According to Cruz et al. matching this type of ontologies is challenging due to poor textual descriptions, flat taxonomy structures, cross-domain coverage and imported ontologies and concepts. In their approach they configured a very high threshold (0.95) for including equivalence relations in the initial alignment since a wrongly determined equivalence relation can propagate errors to all inferred subsumption relations. The performance of the EME approach was evaluated along with other techniques in [22], and precision, recall and F-measure scores are shown in Figure 3.5. As the figure shows, the EME approach is the second best approach in that study, obtaining a precision of around 0.61, a recall of around 0.23 and an F-measure of around 0.35.

3.2.3 Logic-based Techniques

S-Match [46, 133, 48] takes a different approach to semantic matching. S-Match takes two graph structures (e.g. web directories, XML schemas, lightweight ontologies[5]) as input. The core idea of S-match is to represent ontology concepts as well as their context (i.e. a concept along with its path from the root) as propositional logical formulas. These formulas aim to capture the intended meaning of the concepts, represent the concepts using a machine-processable encoding, and thus reduce the matching problem to a propositional validity problem. The overall process includes a set of matchers that on the basis of the logical formulas computes candidate semantic relations between the ontology concepts which are stored in a matrix. There are three categories of matchers: String-based matchers that detect equivalence relations, and sense-based as well as gloss-based matchers that detect subsumption- and disjointness relations. The sense-based matchers and the gloss-based matchers are based on WordNet. In the next step, each relation in the matrix is then checked for validity by proving that the negation of the formula representing this relation is unsatisfiable. This step is performed either by ad hoc reasoning techniques or standard satisfiability solvers.

As part of the S-Match framework, three different algorithms are proposed:

1. The *Basic Semantic Matching* algorithm, which is a general purpose algorithm suited for different graph-based structures and application domains.

2. The *Minimal Semantic Matching* algorithm, which produces a reduced set of relations, but from which all other relations can be computed.

3. The *Structure Preserving Semantic Matching (SPSM)* algorithm, which distinguishes between structural elements (i.e. functions and variables) in the input models. This algorithm is targeted towards API and database schemas.

It should be noted that both Jain et al. [71] and Arnold and Rahm [7] claim that S-Match suffers from low precision due to a very permissive strategy with regards to including relations in the finally produced alignment.

[5] However, the current available implementation of S-Match is not capable of parsing OWL ontologies, as also noted by [7].

3.2.4 Background Knowledge Techniques

Some form of external source of knowledge is usually applied when detecting non-equivalence relations. According to Cruz et al. [22] the use of external lexical resources such as WordNet is crucial when computing subsumption relations.

The *Background Knowledge Strategy* used by Arnold and Rahm [7] in the STROMA system can use different linguistic resources to infer a semantic relation between two concepts, but WordNet is the main resource. This strategy uses the hypernymy sets in WordNet combined with a technique called *Gradual Modifier Removal* to infer subsumption relations. A hypernym is a word with a broader meaning and can be exemplified by 'Color' being a hypernym of 'Red'. The Gradual Modifier Removal technique is based on gradually removing compound modifiers in order to determine a subsumption relation. In their paper, Arnold and Rahm [7] used the following example: when mapping 'US Vice President' and 'Person', the former was not initially not present in WordNet. In the next step of the Gradual Modifier Removal process, the modifier 'US' was removed, and 'Vice President' was found in WordNet. Since WordNet states that 'Person' is a hypernym of 'Vice President' the Background Knowledge Strategy determined that 'US Vise President' is subsumed by (is-a) 'Person'. An evaluation of this strategy was performed on 6 different datasets containing web directories and taxonomies from different application domains. Reference alignments consisting of different semantic relations (including equivalence and subsumption) were created manually for each dataset. As shown in Table 3.2, which is adapted from [7], the background knowledge strategy had a positive contribution to the F-measure scores in most datasets in this evaluation.

Table 3.2: Contribution of the Background Knowledge Strategy in the evaluation of STROMA

	B1	B2	B3	B4.1	B4.2	B4.3
with BK	0.87	0.96	0.87	0.67	0.39	0.43
without BK	0.87	0.94	0.82	0.65	0.31	0.38

As described in the previous section, Giunchiglia et al. [46] include several techniques that exploit the WordNet lexicon in their semantic matching system S-Match. The *sense-based* techniques use the semantic relations between WordNet synsets to derive equivalence, subsumption or disjointness between two concepts. This mapping occurs as follows: If a semantic relation in WordNet is hyponymy resp. hypernymy this results in a isSubsumedBy (<) resp. subsumes (>) relation; if a semantic relation in WordNet between two concepts is synonymy or the two concepts to be matched belong to the same synset this results in an equivalence relation; and finally, if the two concepts are related by an antonymy relation or they are siblings in the part-of (meronymy) hierarchy, the two concepts are considered disjoint.

The *gloss-based* technique is based on the gloss or synset definitions in WordNet. The "basic" WordNet gloss matcher in S-Match compares the label(s) of the first concept C_s with the WordNet gloss of the second concept C_t. This technique is based on counting the number of occurrences of the labels of C_s in the gloss of C_t. First, it extracts the labels of the first concepts from WordNet. If this number exceeds a given threshold, $C_s < C_t$. Another variant of the gloss matcher uses the extended gloss in WordNet, i.e. an aggregation of glosses from a concept's descendants or ancestors. If the gloss of the descendants is used, and the number occurrences of a concept C_s in the extended gloss of C_t is above a threshold, then $C_s > C_t$. Otherwise, if the gloss of the ancestors is used, and the number of occurrences of C_s in the extended gloss of C_t is above the threshold, $C_s < C_t$.

The BLOOMS system developed by Jain et al. [71] is a semantic matching system that computes equivalence and subsumption relations between ontology concepts. Supported by either Wikipedia or WordNet as background knowledge, BLOOMS represents the concepts to be matched as a set of trees, where the trees are composed of related terms retrieved from these external sources. If Wikipedia is chosen as the external source, BLOOMS uses a Wikipedia web service to query Wikipedia articles using the concept names as search terms. Each returned article is considered a *sense* of the concept name and a tree data structure is constructed where this sense is the root and Wikipedia Categories associated with the article represent its children. Hence, for each concept there exists a forest of trees, where each tree represents Wikipedia articles and categories related to the concept. When identifying the relation between two concepts from different ontologies, their respective trees are compared with respect to their overlap. BLOOMS was evaluated using the *Oriented Matching* dataset from OAEI 2009. The evaluation results showed that the BLOOMS approach performed very well in this dataset, with an average precision of 0.84 and an average recall of 0.78 across the three tests.

The BLOOMS system was implemented as one of the semantic matching systems used in a comparative evaluation in this work (see Section 6.1.4). Some observations from this implementation are that the system does not enforce a one-to-one relationship in equivalence relations, the text processing seems very basic (for example, whenever there is a compound word this is treated by just adding whitespace between its parts before querying WordNet/Wikipedia), and the subsumption relations are one-directional (only subsumedBy (less general than)).

The SCARLET system developed by Sabou et al. [125] uses external ontologies in order to find semantic relations between the concepts of two ontologies to be matched. This approach used an ontology search engine such as Swoogle [6] to find relevant external ontologies using the concepts to be matched as queries. In order to "anchor" the concepts in relevant external ontologies SCARLET uses strict string matching, but allows for variations in naming conventions and lexical form (e.g. the lemma associated with a concept's label). Depending on the search results, one of two strategies can be followed: (1) the concepts belong to a semantic relation in one single external ontology. In this case this relation is used for the current alignment being computed by SCARLET, or (2) the concepts form an indirect relation distributed over several external ontologies. In order to detect contradictory and incoherent relations, SCARLET includes a simple alignment debugging mechanism. SCARLET was evaluated using two thesauri as input and an ontology search using Swoogle. A subset of the returned candidate relations was manually assessed by 9 ontology experts. The semantic relations discovered by SCARLET were subsumption (both ways) and disjointness. Some of the key findings from the study were that errors in the anchoring phased accounted for more than 50 percent of the false positive mappings and that subsumption relations were wrongly used to describe some other type of relation between concepts (e.g. part-whole).

Along the same line, Cruz et al. [22] used the fact that ontologies often import other ontologies as a mechanism to infer subsumption relations in the AgreementMaker system. Hence, the source for background knowledge is in this case external linked open data ontologies. This method was called *Global Matching*. For each concept C_s in the source ontology, the method searches across other external ontologies for any candidate concept that has been defined as subclass of C_s. If any identified candidate concept (being a subclass of C_s) exist, this concept is compared with the concepts of a target ontology, and if a concept in the target ontology match, the method return a subsumption relation stating that C_s subsumes the concept in the target ontology. This approach works well for Linked Open Data (LOD) ontologies, that often import multiple other ontologies for describing a particular

[6] http://swoogle.umbc.edu/2006/

domain [22]. The evaluation performed by Cruz et al. also confirms this. The Global Matching approach performed best of the approaches suggested in this study, with a precision of around 0.65, a recall of around 0.35 and an F-measure of around 0.45. A chart of the evaluation scores is presented in Figure 3.5.

Cruz et al. [22] also suggested a technique called Distance-based Polysemic Lexical Comparison (DPLC) in their extension of the AgreementMaker matching system. This technique first annotates the ontology concepts with lexical concepts from WordNet. Then it uses these lexical concepts and how they are positioned in a hierarchy, as well as associated hypernyms, to infer subsumption relations. The confidence score of a subsumption relation is determined on the basis of the path distance between two lexical concepts that annotate the ontology concepts to be matched. Word Sense Disambiguation techniques are applied to maximise the possibility of a lexical concept being semantically equal to a ontology concept. This approach did not perform as well as the *Global Matching* approach described above. As seen in Figure 3.5, the DPLC approach obtained a precision of around 0.32, a recall of around 0.11 and an F-measure of around 0.16.

In a recent paper, Kamel et al. [79] reports on the use of BabelNet, a semantic network that among other sources exploit WordNet and Wikidata, to identify subsumption relations between ontologies. The approach followed includes a two-step process. In the first step, the concepts to be matched are disambiguated by identifying the semantically closer synset in BabelNet. This is accomplished by first creating a context for the concept and a context for the associated BabelNet synset. The context for the concept is represented using the label, super- and subclasses, etc., whereas the context for the BabelNet synset is constructed using their sense and gloss terms. Then, a set of tokens (bag-of-words) is created by finding the overlap between these two contexts, which finally represents the concept. In the second step, the algorithm looks for subsumption relations between two concepts that are represented by the set of tokens established in the first step. This is accomplished by checking if the tokens representing a source concept reside in the set of hypernyms of tokens representing the target concept and vice versa. If so, a subsumption relation between these two concepts is derived. The approach was evaluated using inferred subsumption alignments from the equivalence alignments of the OAEI conference dataset. The evaluation results showed in overall a low performance, with an average precision of 0.29 and an average recall of 0.11 when isolating the scores on three pairs of ontologies (*edas-ekaw*, *confOf-edas*, and *conference-sigkdd*). This was according to the authors due to lack of coverage in BabelNet and lack of annotations in the ontologies being matched resulting in sparse context descriptions.

3.2.5 Natural Language Processing Techniques

Various Natural Language Processing (NLP) techniques are used in semantic matching as supportive means to other techniques.

Po and Bergamaschi [114] enhanced the SCARLET system mentioned in the previous section by lexically annotating concepts using a combination of Word Sense Disambiguation (WSD) techniques. This lexical annotation process aimed to improve both precision, in that false positives could be identified and removed from the resulting alignment, and recall, in that additional semantic relations could be discovered and included in the resulting alignment. Figure 3.7 shows an existing subsumption relation between concept A in Ontology 1 and concept B in Ontology 2. This relation is derived since SCARLET has identified that (1) A and A' as well as B and B' are idenfied as equivalent, and (2) the same subsumption relation is found to exist between concepts A' and B' in the online ontology. Since A and A' can be linked to the same WordNet synset (Synset 2) according to the disambiguation performed, these two concepts are considered semantically equivalent. However, since concepts B and B' are linked to separate WordNet synsets after having run some disambiguation technique, these two concepts are considered semantically different, and the original subsumption relation between A and B is therefore considered errouneous.

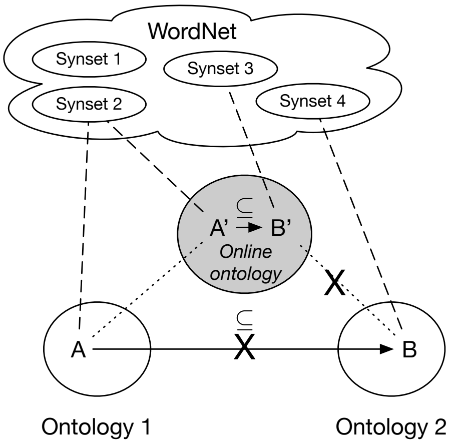


Figure 3.7: Using Lexical Annotation to enhance SCARLET (adapted from Po and Bergamaschi [114]).

The evaluation of the lexical annotation approach showed that the lexical annotations could improve both precision in that false positive relations could be omitted and recall since additional true positive relations were identified.

As Falcon-AO described earlier in Section 2.5.4, TaxoMap [60, 58] targets large ontologies and uses a partitioning approach that decomposes large input ontologies into blocks from which semantic relations are computed. TaxoMap computes alignments consisting of equivalence, subsumption and semantically related relations. In order to arrive at these relations, TaxoMap uses linguistic and structural techniques. Here, we will focus on how Taxomap uses linguistic techniques to infer semantic relations. These techniques rely on a decision tree tagger [127] that analyses the part-of-speech (POS) and lemma information of labels. This tagger distinguishes full words from complementary words based on whether they are nouns or functional words (verbs, adverbs and adjectives), and how the words are positioned in the corresponding labels. When identifying subsumption relations, one of the matchers ($LabelInclusion$) in Taxomap uses the following heuristics to determine that c_s in ontology O_S is subsumed by c_t in ontology O_T (i.e. that c_s is less general than c_t):

- c_t is the concept label in O_T having the highest similarity value (based on trigrams) with c_s.
- One of the labels of c_t is *included* in the label of c_s.
- All the words of label of c_t are classified as full words by the tagger.

TaxoMap was evaluated in a dataset involving two ontologies describing geographical concepts [59]. Equivalence and subsumption relations, as well as semantic closeness relations, were identified using the abovementioned techniques and comparing two different partitioning strategies. The results from the evaluation showed that TaxoMap obtained precision/recall scores of 0.97 and 0.81 respectively using the best partitioning strategy.

3.2.6 Machine Learning Techniques

The usefulness of machine learning strategies, both supervised and unsupervised strategies, has also been explored when automatically identifying subsumption relations.

Spiliopoulos et al. [137, 136] used a supervised machine learning scheme and considered the identification of subsumption relations as a binary classification problem. They called this method *Classification-Based Learning of Subsumption Relations (CSR)*, and it consists of the following steps: (1) Infer internal subsumption relations within each input ontology using a reasoner, (2) Generate classifier features from common words, properties and latent features approximating the intended meaning of concepts, (3) Generate training examples for the subsumption class and for the non-subsumption class. For the subsumption class training examples are generated by considering all subsumption relations inferred in (1) and optionally from subsumption relations inferred from equivalence relations identified by equivalence matching. The training examples for the non-subsumption class are generated from siblings sharing the same subsumer in each ontology, concepts that are explicitly not in a subsumption relation, or inverse pairs of concepts that are related with a subsumption relation, (4) Train the classifier using the training examples generated in the previous step, (5) Classify each relation as either a subsumption relation or as a non-subsumption relation. The evaluation of the CSR method was based on two datasets from OAEI as well as a dataset based on course catalogues from Washington and Cornell Universities. Since there were no existing datasets for subsumption matching, subsumption alignments were generated from original equivalence alignments as follows: (1) Infer subsumption relations from equivalence relations using a reasoner, (2) Extend the set of subsumption relations from (1) by using common sense based on understanding the "intended meaning" of the concepts in the input ontologies. The evaluation of the CSR approach showed that it was able to locate subsumption relations that cannot be inferred from existing equivalence relations using a reasoner, it was able to discriminate between subsumption relations and equivalence relations, and that the C4.5 decision tree classifier performed better than the other classifiers tested (Knn, Naive Bayes and SVM), especially for well-annotated concepts.

David et al. [74] developed an approach for identifying both equivalence and subsumption relations using *association rules* [1] in the AROMA system. Association rules are typically used in data mining, and a common example used to explain the association rule paradigm is the "market basket analysis". Here, the aim is to discover purchase patterns from transactional data from supermarkets: From an analysis of these data an association rule algorithm (e.g. the apriori algorithm [2]) suggests the following association rule (borrowed from [90]):

Cheese –> Beer [support = 10, confidence = 80].

This rule states that 10 % of the customers buy cheese and beer together, and those who buy cheese also buy beer 80 % of the time. Support and confidence in this example are so-called *Interestingness Measures (IMs).*

David et al. propose a new interestingness measure based on implication intensity (e.g. described in [50]) that targets the automated detection of subsumption relations and equivalence relations. This approach is based on the assumption that a concept C_s will be more specific or equivalent to a concept C_t if the vocabulary used to describe C_s, its sub-concepts, and its instances tend to be included in that of C_t, under some conditions expressed by the new interestingness measure. Vocabulary in this context means the name, definitions and instances associated to a concept. The matching process proceeds as follows:

1. Extract a set of the relevant terms for each concept and property.
2. Discover association rules using an interesting measure based on implication intensity between entities based on the respective sets of relevant terms.
3. Enhance the proposed alignment by removing redundant correspondences and adding additional relations not discovered by the previous process. Here, the approach uses the Jaro-Winkler [156] string similarity technique.

AROMA was evaluated using the OAEI benchmark from 2005. The results of the evaluation showed that the approach favors precision over recall. Considering the harmonic mean of the three tracks that were included in the evaluation, AROMA obtained a precision of 0.96, the highest score of all compared systems. The recall achieved by AROMA (0.6) was however lower than the results obtained by a basic string matcher based on edit distance.

3.3 Profiling Ontologies

By profiling we mean quantifying certain features from the ontologies that will guide different steps of the matching process. The profiling can use criteria and metrics from related research areas ontology analysis and ontology evaluation. Existing literature on ontology analysis and ontology evaluation is extensive, and a number of surveys offering a comprehensive overview have been conducted on the topic [44, 116, 155, 11].

The typical objective of ontology analysis is to assess the quality of ontologies for the purposes of reusing- or improving them. On this account, Noy and Hafner [106] developed in an early work a framework for comparing ontologies according to eight overall characteristics. The characteristics are: *general* (i.e. what purpose is the ontology developed for), *design process* (i.e. how was the ontology developed), *taxonomy* (e.g. how is the ontology structured), *internal concept structure and relations between concepts* (e.g. how and to what extent are properties implemented in the ontology), *axioms* (e.g. how are the axioms expressed), *inference mechanisms* (e.g. how is the reasoning done), *applications* (i.e. what practical application was the ontology intended for), and *contributions* (e.g. is it multilingual, is it authoritative in its domain, etc.).

Gangemi et al. [43, 42] suggests a framework comprising three types of measures for analysing ontologies: *Structural measures*, which considers the graph-based characteristics of the ontology; *functional measures*, which assess to what extent the ontology models the purpose for which it was developed; and *usability-profiling measures*, which relates to how well the ontology and its constructs are described through annotations, metadata, and other documentation.

In order to analyse the ontology schema, which is most relevant in our analysis, Tartir et al. [146] propose three metrics: *Relationship Richness*, which reflects the diversity and placement of object properties in the ontology; *Attribute Richness*, which through computing how many attributes (data properties) exist for each class gives some insight into how much knowledge is expressed by classes in the ontology; and *Inheritance Richness*, which gives an indication of how well knowledge is grouped into different categories and subcategories in the ontology.

Some of the metrics suggested by Tartir et al. are implemented in the OntoMetrics tool [85]. This is an ontology analysis tool comprising a large number of different metrics for ontology analysis. Another ontology analysis tool is the Oops validation tool [118]. The Oops tool validates if an ontology conforms to best practices in ontology engineering by automatically checking if the ontology has managed to avoid common errors or pitfalls that might occur when developing ontologies.

When it comes to using ontology features to support matching operations, the RiMOM system [87] employs two *similarity factors* that quantitatively characterise the ontologies to be matched. The first is called *Label Similarity Factor* and aims to define the similarity between two ontologies based on entity names. This is computed by taking the sum of number of identical class labels and the sum of identical property labels over all concepts and properties in the two ontologies to be matched. The second measure is called *Structure Similarity Factor* and provides a quantified characteristic of both classes and properties. First, all classes that have subclasses associated with them are compared. For each pair of classes that have the same number of subclasses and the same path length to root, the variable *#comm_nonl_conc* is enumerated. The *Structure Similarity Factor* is normalised by dividing the *#comm_nonl_conc* for ontology O_s with the *#comm_nonl_conc* for ontology O_t.

Cruz et al. [21] extracts a set of ontology features to be used in supervised machine learning (described in Section 3.4). These features are:

- *Relationship Richness*, this metric is defined as the percentage of object properties that are different from subClassOf relations.
- *Attribute Richness* is defined as the average number of datatype properties per class.
- *Inheritance Richness*, this metric is a structural characteristic and is defined as the average number of subclasses per class.
- *Class Richness* is defined as the ratio of classes having instances associated with them.
- *Label Uniqueness* is computed as the percentage of concepts that have a label that differs from the concept name.
- *Average Population*, this metric is defined as the number of instances divided by the number of classes in an ontology.
- *Average Depth*, this metric captures the average depth of the classes in an ontology and is calculated as the mean of the depth over all classes.
- *WordNet Coverage* is computed as the percentage of concepts with a label or URI present in WordNet.
- *Null Label and Comment*, this metric is computed by dividing the number of concepts that have no comment or label over all concepts in an ontology.

The three first metrics in the list above are defined by Tartir et al. [146].

In the UFOMe [113] system two profiling metrics are applied in order to select, configure and combine individual matchers. These metrics are called *lexical affinity* and *structural affinity*. The lexical affinity basically is the fraction of lexically similar entities / the total number of concepts in the smallest ontology. The structural affinity is based on information content (IC) where the IC of a concept is expressed according to its position in the class hierarchy. The structural affinity is then calculated as the fraction of the number of entities having a similar IC / total number of concepts in the smallest ontology.

3.4 Matcher Selection, Matcher Configuration and Alignment Combination

This section begins by describing different approaches to automated matcher selection and configuration. After that, we describe different approaches for combining multiple candidate alignments into a final alignment. Different matchers provide different results depending on the characteristics of the ontologies to be matched [35], and identifying the optimal set of matchers for a given matching task is considered one of the top challenges of ontology matching [131, 132]. Furthermore, normally, each matcher in a matching system can be configured with a threshold that reflects how confident the matcher is that a given relation in an alignment is correct. This confidence threshold is usually also used as a weight when alignments from several matchers are combined. Setting this confidence threshold is thus of critical importance to the alignment quality as setting it too low will likely result in false positives, and setting it too high will likely result in false negatives.

Mochol et al. [100, 99] describes how an analysis of metadata associated with ontologies combined with a description of matchers made by their developers can guide the selection of relevant ontology matching techniques. The ontology metadata includes typical ontology statistics (size, formality level, natural language level, etc.). The matcher descriptions include details such as specific techniques applied (e.g. string matching), usage characteristics (e.g. if the matcher targets a particular application domain), cost characteristics (licensing costs) and alignment constraints (e.g. 1-1 class relations). A set of SWRL rules are applied to identify the most appropriate set of matchers for a given pair of ontologies to be matched. A limitation of the approach by Mochol et al. is that it assumes a rich and accurate description of each matcher in order to find a good fit between the ontologies to be matched and matcher capabilities.

Tan and Lambrix [144] proposed an approach for selecting matchers based on how they perform on a subset of the ontologies to be matched. This semi-automatic approach relies on evaluating the results when an ensemble of matchers perform a matching of this ontology subset in comparison with a reference alignment that is already existing or that is evaluated by manual analysis.

Cruz et al. [21] performs supervised machine learning (k-NN) to select the optimal matcher configuration from a set of pre-configured composition of matchers given a profile of the two input ontologies. The profile is established by using the set of ontology evaluation metrics described in Section 3.3. The resulting alignments produced by the matchers involved in a particular matching operation are combined by linearly weighting the individual alignments. In the learning phase a subset (20 %) of the dataset is used as training set in order to choose the best matcher configuration for this dataset. Then the subset is being matched in parallel and the resulting alignments from the different matcher configurations are compared to the reference alignment for that particular subset. The approach does not focus on selecting individual matchers from a library, but selecting a pre-defined matcher composition from the AgreementMakerLight matching system (see Section 3.1) given a set of ontologies.

The approach by Cruz et al. requires training data in the form of a set of correct relations defined by human judgment. This implies that human effort is required in order to establish the training data for each matching task. Furthermore, they use predefined configuration of the matchers, that is, the similarity thresholds used by the different matchers are fixed regardless of the ontology characteristics in each matching task.

The RiMOM matching system [87] extracts terminological and structural profiles of the ontologies to be matched. These profiles are then used to automatically select the information to be used by the terminological matcher, how to weight the relations computed by the different matchers and the similarity propagation strategy used by RiMOM. If for example the profiling of the input ontologies reveals a high structural similarity factor, the vector-based matcher includes structural information (e.g. number of subconcepts) associated with the concepts being used when constructing the TF-IDF vector, if not, such information is omitted. The thresholds used to determine whether the lexical similarity factor and the structural similarity factor should be considered high or low are based on experimentation.

In the following we describe different approaches for combining alignments produced by multiple matchers into a final alignment. Normally the matching process involves an alignment combination / aggregation step that aims to aggregate the optimal relations from alignments produced by the involved matchers into a final, refined alignment. Combining the results from individual matchers can often improve the final alignment quality [112]. A large variety of combination methods exist, some are basic and others more advanced. According to a study by Peukert et al. [112] advanced combination methods can perform well on some matching tasks, but in general simpler methods, such as using Average Aggregation (i.e. using the average confidence score from the individual matchers for each relation in the final alignment) are more robust. Indeed, some of the best performing matching systems, such as AgreementMakerLight [39] and COMA [25, 93] utilise quite basic methods when combining the results from individual matchers.

In this section both simple and advanced combination methods will be described. Note that some of the more advanced combination methods use machine learning techniques, such as the method proposed by Eckert et al. in [27]. However, since these techniques require training data in the form of ground truth alignments which are usually not available [120, 112, 91] thus making these techniques inapplicable, they are not described any further here.

Of basic methods commonly described in literature we find *average*, *max*, *min*, *threshold* and *delta*. The *average* method computes an average similarity over all individual matchers that have identified a given relation. This means that all matchers are considered equally important [25]. The *max* method returns, for a given relation, the highest similarity value of any individual matcher. This is a very optimistic approach since a relation that is only proposed by one single matcher in an ensemble can make it through to the final alignment [91]. In the opposite end, the *min* method chooses the lowest similarity value for a given relation from any individual matcher, and is as such considered a very pessimistic approach. There are several variants of the *threshold* approach, but in its basic form this implies that a predefined cut threshold determines which relations will be included in a final alignment [35]. For example, if a threshold is set to 0.6, only those relations from the individual matchers that have a confidence value above or equal to 0.6 will be included in the final alignment. The *delta* is a variant of the threshold approach whereby the relations that have the highest confidence value above a predefined threshold are included in the final alignment as well as the set of relations that have a confidence value that falls within a predefined distance relative to the top relations [93].

An overview of some more advanced combination methods is provided in the following. Trojahn et al. [148] suggests a multi-agent approach to ontology matcher combination. Here, one agent plays the role of mediator and three other agents play the role of lexical matcher, semantic matcher and structural matcher. These agents all compute their separate alignments and after a negotiation phase a final alignment is produced. During the negotiation, correspondences not verified by one agent (i.e. the score is below a certain threshold) are distributed to the other agents by the mediator. The other agents either confirm the correspondence or make a counter proposal and the process iterates until the matcher agents run out of counter proposals.

The UFOme [113] matching system includes a *Strategy Predictor* module that automatically selects, configures and combines individual matchers based on terminological and structural ontology characteristics (see Section 3.3 for details on how these ontology characteristics, denoted *affinity values* in the paper, are defined). UFOme includes four individual matchers, of which three of them are terminological/lexical matchers and the fourth is a structural matcher. The affinity values computed in the ontology profiling process are used by the Strategy Predictor to determine the confidence thresholds and the weights associated with the individual matchers. The rationale used for determining the confidence thresholds is that if the affinity scores are high, the thresholds should be lower. The rationale used for setting the weights is that if the affinity score is high, the weight for the associated matchers should be high. The smoothing factors used to determine the thresholds and weights were decided through experiments using a subset of ontologies in the evaluation dataset. Specifically, confidence thresholds of 0.6 and 0.4 were used as parameters for the smoothing factors. The combination of the individual matching results is based on weighing the individual alignments and combining them using a weighted sum. The evaluation of the Strategy Predictor module showed the importance of setting the confidence thresholds and weighting parameters correctly. In the worst case, the alignment quality decreased by 15% in terms of precision and 20% in terms of recall when a sub-optimal threshold was applied.

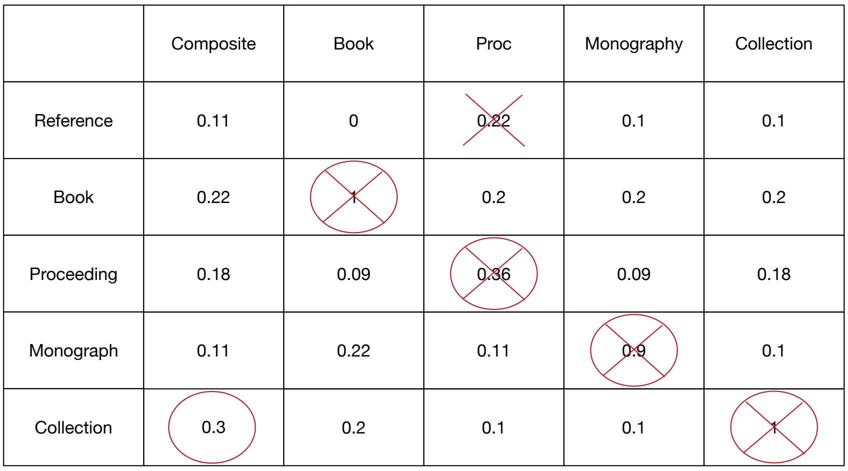
Cruz et al. [20] implemented a method for combining the results from individual matchers in the AgreementMaker matching system. This method is called *Linear Weighted Combination (LWC)*. When producing the final alignment, the relations produced by the individual matchers are combined by taking into account the weights associated with these matchers. The weights associated with the individual matchers were assigned automatically based on the notion of a *Local Confidence* of a matcher. The Local Confidence is a measure that basically extracts the average similarity value of those relations that are not selected (below an arbitrary threshold) from the average similarity value of those relations that are selected (above an arbitrary threshold) from a matrix of relations produced by that matcher.

The Autoweight++ method [56] includes both a matcher configuration and combination approach. The concept of extracting *highest correspondences* from a similarity matrix (see Figure 3.8) is central in this approach. A correspondence between two entities e_i and e'_j is considered the highest correspondence if it has a higher confidence value than any other correspondence that includes either e_i or e'_j. A highest correspondence threshold determines which of the considered highest correspondences are processed further. The highest correspondences are first computed for each individual alignment. Then, for every alignment produced by all matchers, an importance coefficient for each highest correspondence is computed. This importance coefficient considers how many matchers have identified this particular correspondence. So the importance of each particular highest correspondence is based on how many times this correspondence has been detected as a highest one across all correspondences from all matchers. If the highest correspondence is identified by all matchers (i.e. all alignments), it is omitted since it brings no useful and discriminating information.

The matcher weight, which is also used in the combination, is set based on the importance of the correspondences it produces compared with the other matchers' correspondences. So the importance (coefficient) for a matcher is calculated by summing the importance values of all highest correspondences produced by that matcher. The weight of a basic matcher is the ratio of the importance coefficient for that particular matcher and the sum of the importance coefficient of all matchers. When aggregating all correspondences from all matchers, the aggregated correspondence for two entities is calculated by multiplying their correspondence strength in each alignment (from each matcher) with the weighting factor (assigned to the matcher/alignment) and summing up those products.

Similar to Autoweight++, the Harmony-based Adaptive Similarity Aggregation (HADAPT) method suggested by Mao et al. [91] starts by representing all relations computed by a single matcher in a similarity matrix. From this matrix the relations that have the highest similarity value across rows and columns are used to compute the so-called harmony value of each individual matcher. This is illustrated in Figure 3.8 where the highest similarity values in each row is indicated by a cross and the highest similarity value in each column is indicated with a circle. The harmony value is the ratio between the number of relations with the highest similarity value and all possible relations from the two ontologies being matched. In the example there are

5 suggested relations, and four relations where the similarity value is highest in both row and column. This yields a harmony value of 0.8 ($\frac{4}{5}$). When combining the alignments from all individual matchers, the harmony value is used to weight the different matchers.



	Composite	Book	Proc	Monography	Collection
Reference	0.11	0	0.22	0.1	0.1
Book	0.22	1	0.2	0.2	0.2
Proceeding	0.18	0.09	0.36	0.09	0.18
Monograph	0.11	0.22	0.11	0.9	0.1
Collection	0.3	0.2	0.1	0.1	1

Figure 3.8: Similarity matrix (adapted from Mao et al. [91]).

When combining the results from the individual matchers, STROMA uses predefined weights for each matcher based on experimental results and a voting strategy to conclude the type of semantic relation. The Compound Strategy, Background Knowledge Strategy and Itemization Strategy all have a weight of 1.0. The Structure-based Strategy has a weight of 0.8, and the Multiple Linkage Strategy has a weight of 0.5. If all strategies return "undecided", the relation type is set to equivalence, while if two or more matchers propose conflicting semantic relations at the same confidence, a predefined priority order determines the final relation type. The priority order is: equivalence, subsumption, meronymy, semantically related.

Part II

Research Approach

4 Use of Design Science to Address the Research Objectives

The research approach is inspired by principles from Design Science [63]. Design Science involves rigorous and iterative creation and evaluation of innovative artefacts. These artefacts embody ideas, practices, technical capabilities, and products seeking to address concrete problems and needs from the application domain in question. Furthermore, the creation and evaluation of the artefacts should be conducted rigorously, benefiting from existing relevant knowledge and offering useful knowledge in return.

Design science fits well with the research conducted in this thesis as it prescribes and guides a rigorous approach to information systems artefact development. Although the research challenges dealt with in this thesis are more positioned in basic research than applied research, they are sufficiently well-defined (and well-acknowledged) to be supported by the guidelines prescribed by the design science framework. There are several research gaps within semantic matching worth pursuing and the research conducted in this thesis aim to cover some of them. Addressing some of these gaps can likely support a transition to a more applied type of research and higher levels of the Technology Readiness Level (TRL) scale. The utility of the artefacts produced is diverse as semantic matching is relevant in a wide range of tasks. Many of which today require significant human labour that could be applied to other core tasks within the software engineering lifecycle.

Figure 4.1 shows the design science framework [65] adapted to the work in this thesis. From the top-left, problems and opportunities derive from the environment in question. These problems and opportunities are the starting point for the design science research, and they can be sourced from people, organisations and the technological level. Ideas of artefacts that can either address the problems or realise the opportunities can emerge when the problems and opportunities are explicated and understood. This triggers a build-and-evaluate loop where artefacts are developed and evaluated using rigorous development- and evaluation methods. The development and evaluation activities are supported by existing knowledge in the knowledge base (to the right). Intermediate and final experiences and results from this cycle are fed back to the knowledge base as novel research contributions and to the environment as applications satisfying the initial problems/opportunities.

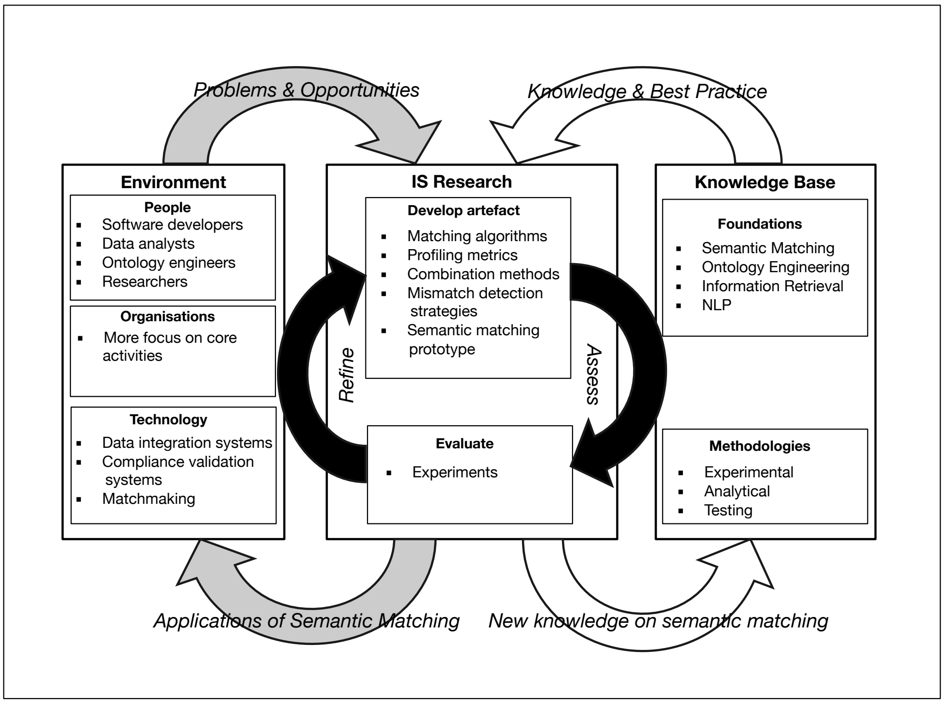


Figure 4.1: Design Science Framework.

Hevner [63] stresses that a design science research project should be guided by three interdependent cycles.

1. Relevance Cycle: This cycle puts requirements from the contextual environment into the research, and as output from the research, it introduces the research artefact(s) to the environment. This cycle is represented by the gray arrows in Figure 4.1.

2. Rigor Cycle: The rigor cycle provides theories and methods along with domain experience and expertise from the knowledge base into the research. It adds the new knowledge generated by the research to the growing knowledge base. This cycle is represented by the white arrows in Figure 4.1.

3. Design Cycle: Based on input from the other two cycles, the creation, and evaluation of the artefact(s) is conducted in rapid iterations. This typically involves generating candidate designs and ideas which are assessed and refined until a satisfactory solution is achieved (Simon [134], cited in Hevner [65]). This cycle is represented by the black arrows in Figure 4.1.

The Design Science framework further specifies seven research guidelines that should be addressed when conducting design science research. These seven guidelines are presented in the following with an explanation on how each of them has been addressed by the work in this thesis.

4.1 Design as an Artefact

> "*Design-science research must produce a viable artefact in the form of a construct, a model, a method, or an instantiation.*"

According to Hevner et al. (2004) [65], there are four categories of artefacts:

- Constructs: This type of artefacts provide the language in which problems and solutions are defined and communicated.
- Models: This type of artefact use constructs to represent a real-world situation - the design problem and its solution space. An example of a model artefact is a system architecture that uses notation and symbols constructs to describe a particular context.
- Methods: These artefacts define solution processes. They can range from formal, mathematical algorithms to textual descriptions of best practice approaches.
- Instantiations: This artefact type shows how to implement constructs, models, or methods in a working system. They demonstrate feasibility and enable detailed assessment of an artefact's suitability to its intended purpose.

Based on the problems and needs from the environment in which the artefacts will be used, and by using relevant existing knowledge, researchers build theories and artefacts which are evaluated using appropriate evaluation techniques. The evaluation generates feedback that improves the researcher's understanding of the problem and the artefacts' ability to address it. This leads to refinements of the theories and artefacts. This build-and-evaluate loop is often performed iteratively, improving theories, artefacts as well as the design processes in each iteration.

In this work, the primary focus is on developing method artefacts and instantiation artefacts. The method artefacts are materialised as matching algorithms, ontology profiling metrics, strategies for combining alignments, and strategies for detecting mismatches. Finally, the semantic matching prototype, which brings together all the other artefacts into a complete system, represents an instantiation artefact. An overview of these artefacts including a short description, which artefact type they represent as well as how they relate to the research questions in this thesis is presented in Table 4.1.

4.2 Problem Relevance

> "*The objective of design science research is to develop technology-based solutions to important and relevant business problems.*"

Having worked as an applied researcher for over a decade, the candidate has worked closely with interoperability challenges together with the industry. In very many research projects there is a recurring challenge that in order to realise or demonstrate some innovation there is a cumbersome and resource-intensive phase that needs to be conducted before-hand. This is the "mapping-phase" where two, and often more than two, exchange formats, taxonomies, ontologies, databases, specifications, etc., need to be manually aligned element-by-element before the core activities can commence. Supportive tools that can reduce this manual effort would allow an increased focus on the core business and/or research innovations due to time and resources saved on "mapping-phases".

Table 4.1: Overview of artefacts

Artefact	**Description**	**Artefact Type**	**Addressed RQ**
Ontology Profiling Metrics	These metrics aim to support the autonomy of the semantic matching system from an analysis of the ontologies being matched.	Method	RQ1
Matching Algorithms	Matching algorithms identifying both equivalence and subsumption relations between ontology concepts.	Method	RQ2
Alignment Combination Methods	Methods combining individual alignments produced by matching algorithms.	Method	RQ3
Mismatch Detection Strategies	Strategies increasing the precision of the alignments returned from the semantic matching system.	Method	RQ4
Prototype of Semantic Matching System	Prototype integrating all other artefacts into a semantic matching system.	Instantiation	All

As mentioned in the introduction semantic matching as a utility has many application areas, including semantic interoperability (in general), data integration, semantic matchmaking and compliance validation. A few business cases from European and National research projects that highlight the actuality of semantic matching are presented in the following.

Semantic Interoperability

Semantic matching techniques can support semantic interoperability among interacting information systems through the automated or semi-automated declaration of similar or related concepts or format elements in the interacting systems. One concrete example of this is the EU project Digital Water City[a]. In this project, the overall goal is to digitalise urban water management in Europe. This involves among other things to monitor water quality with a variety of sensors, couple this data with data from other sources (e.g. weather and climate forecasts), distributing this data between a large number of partners across Europe that likely have different systems and formats. This is a tremendous interoperability challenge. The project develops a semantic interoperability mechanism whereby ontologies (using multiple ontologies from the water management domain, but also more general ontologies) constitute a common vocabulary and can provide "semantic translation". Semantic matching will be a valuable utility to identify the relations that exist between this set of ontologies. However, as the majority of current semantic matching systems only return a list of equivalence relations, there is still much manual effort required to identify other relevant relations, such as subsumption relations that is one of the focus areas in this thesis (RQ2).

[a] https://www.digital-water.city

Data Integration

During data integration an important prerequisite is to map schemas and data items e.g. to ensure consistency and avoid redundancy. Typically, such a mapping task is performed manually with a significant cost. An earlier study reported by Halevy [57] concluded that during typical data integration projects mapping the data sources represented over half the effort (sometimes up to 80 %). The Norwegian research project ReiseNavet aims at developing a National platform for supporting the construction of Mobility as a Service (MaaS) services in Norway. For this to happen, data must be integrated from numerous sources (transport service providers, added-value service providers, payment operators, ticketing, etc.). The conceptual and physical data models must be aligned before such integration of data can be realized. This alignment process can be supported by semantic matching.

Semantic Matchmaking

Matchmaking is considered a type of retrieval activity that uses different data properties from different sources to match a request for *something* and the best matching result representing this very same *something*. Think of matchmaking engines such as match.com that matches people based on their expressed properties. In the EU project MANU-SQUARE[a] the aim is to develop a marketplace for the exchange of manufacturing resources (e.g. manufacturing processes and equipment). Here, customers having a need for a manufacturing service can identify one or more optimal suppliers that may service that need. In MANU-SQUARE, ontologies are used for describing manufacturing concepts such as processes, equipment, materials, stakeholder types, and so on. The customer request is annotated using concepts from the MANU-SQUARE ontology and so are the resources offered by suppliers. As part of the process of matching customers and suppliers, semantic matching plays an important role. Often, there is no exact match (equivalence) between the concepts (e.g. a process) sought by the customer and concepts representing the offer from suppliers. Hence, a system that is only capable of identifying equivalence and not asymmetric relations such as subsumption would miss sub-optimal matches that still may be useful in this case. For example, a customer may be interested in suppliers that can offer a milling process. The customer's query is formalized using concepts from ontology O_S. Supplier resources are formalized using concepts from ontology O_T. At the moment there are no suppliers that have expressed that they can perform “Milling” in the marketplace, but there are suppliers that have stated that they offer “Precision Milling”. “Precision milling” is a sub-class to “Milling” in O_T. A subsumption matcher could identify these suppliers as potential matches to the consumer query.

[a] https://www.manusquare.eu/

Compliance Validation

The EU project BEST[a] looked at how semantic technologies could be used to improve information exchange in Air Traffic Management (ATM). One of the cases in this project dealt with compliance validation. Compliance validation is an important task that ensures that all information exchange formats used in relation to ATM comply semantically with the standard reference information model in this domain, ATM Information Reference Model (AIRM) [31]. There is a specification that defines a set of criteria for compliance, and these criteria include among other:

- Rewritten: it is acceptable for a data element in the information exchange format to have another name, but it should be semantically equal to the corresponding element in AIRM.
- Restriction: it is acceptable for a data element in the information exchange format to be more restricted (add additional qualifiers) than a corresponding element in AIRM.
- Generalised: it is not acceptable for a data element in the information exchange format to be more general than a corresponding element in AIRM.

The BEST project wanted to see if the compliance validation task could be automated and developed matching algorithms for this purpose. The AIRM was transformed from its original UML format to OWL and as were the information exchange standards. Equivalence matching algorithms were used to detect the "Rewritten" criterium, while subsumption matching algorithms were used to detect the "Restriction" and "Generalised" criteria. Experiences from this work are reported in Vennesland et al. [150].

[a] https://project-best.eu/

The above examples have described a need for semantic matching from a business and applied research perspective. From a basic research perspective Shvaiko and Euzenat [131] elicited what they thought of as the ten topmost challenges in ontology matching. One of these challenges is "Matcher Selection" and "Self-configuration". Sub-challenges sorting under this heading include "Matcher Selection", "Matcher Combination" and "Matcher Tuning", all challenges being addressed as research questions of this thesis.

4.3 Design Evaluation

> "*The utility, quality, and efficacy of a design artefact must be rigorously demonstrated via well-executed evaluation methods.*"

To evaluate the artefacts developed, an experimental research strategy [107] was applied. The research model used for the experimental evaluation is presented in Figure 4.2.

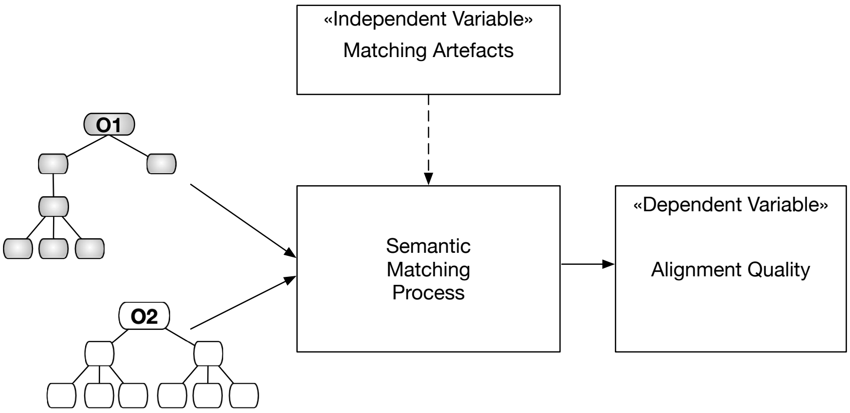


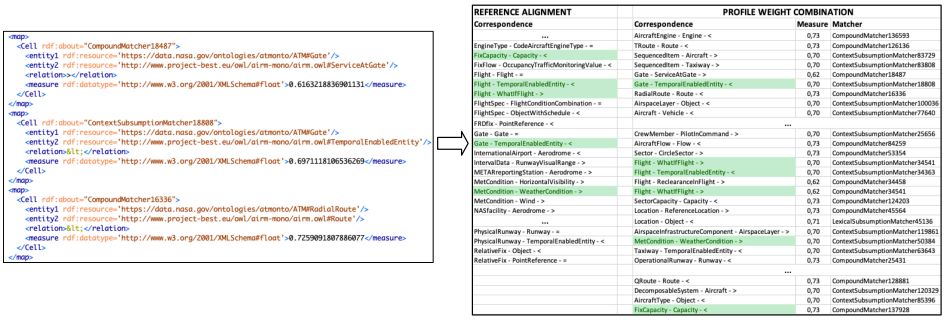
Figure 4.2: Research Model including Independent and Dependent Variables.

Here, the independent variable, that is, the variable that is experimented with, is represented by the matching artefacts developed in this work. These influence to what extent the Semantic Matching Process is capable of producing a good quality alignment as a result. The dependent variable, the variable representing the measured outcome of the experiments, is represented by the quality of the final alignment holding a set of correct semantic relations between two ontologies as input.

The performance of all developed matching artefacts (the independent variable) is evaluated using typical evaluation metrics used in the application domain, namely precision and recall, semantic precision and recall, and F-measure (see a description of all evaluation measures in Section 2.2.3).

These evaluation measures are applied as follows: When comparing the performance of the individual artefacts used in the composition of the semantic matching prototype (the instantiation artefact), standard precision, recall and F-measure is applied as evaluation measures. These evaluation measures are also used when comparing the performance of the semantic matching prototype against other semantic matching systems producing both equivalence and subsumption alignments. This is a valid approach since there exist complete reference alignments that include both equivalence and subsumption relations in all three datasets used in the evaluation. When comparing the performance of the prototype semantic matching system with other matching systems that only produce equivalence alignments, semantic precision and recall are applied as evaluation measures. The reason for using semantic precision and recall is that these measures consider inferred subsumption relations from the closure of the produced equivalence alignments, something which enables a valid comparison with these systems.

In addition to the statistical measures of quantitative data, a comprehensive manual analysis of the resulting alignment is performed. In this alignment analysis the produced alignments are transformed into a more reader-friendly tabular format. This format better facilitates an analysis of how relations in the new alignment compares with relations produced by previous versions of this technique as well as the relations in the relevant reference alignment. Initially, these alignments are formatted as RDF/XML documents (as presented to the left in Figure 4.3), in some cases containing tens of thousands of relations. In order to make such analysis more comprehensible, a set of tools were developed that provided a more user-friendly representation. These simple tools used XSLT-scripts to transform from the RDF/XML representation to an XML format that could be viewed in Microsoft Excel for a more approachable analysis. An example is shown in Figure 4.3.



```
<map>
  <Cell rdf:about="CompoundMatcher18487">
    <entity1 rdf:resource='https://data.nasa.gov/ontologies/atmonto/ATM#Gate'/>
    <entity2 rdf:resource='http://www.project-best.eu/owl/airm-mono/airm.owl#ServiceAtGate'/>
    <relation>></relation>
    <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>0.6163218836901131</measure>
  </Cell>
</map>
<map>
  <Cell rdf:about="ContextSubsumptionMatcher18808">
    <entity1 rdf:resource='https://data.nasa.gov/ontologies/atmonto/ATM#Gate'/>
    <entity2 rdf:resource='http://www.project-best.eu/owl/airm-mono/airm.owl#TemporalEnabledEntity'/>
    <relation>&lt;</relation>
    <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>0.6971118106536269</measure>
  </Cell>
</map>
<map>
  <Cell rdf:about="CompoundMatcher16336">
    <entity1 rdf:resource='https://data.nasa.gov/ontologies/atmonto/ATM#RadialRoute'/>
    <entity2 rdf:resource='http://www.project-best.eu/owl/airm-mono/airm.owl#Route'/>
    <relation>&lt;</relation>
    <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>0.7259091807886077</measure>
  </Cell>
</map>
```

REFERENCE ALIGNMENT	PROFILE WEIGHT COMBINATION		
Correspondence	Correspondence	Measure	Matcher
...	AircraftEngine - Engine - <	0,73	CompoundMatcher136593
EngineType - CodeAircraftEngineType - =	TRoute - Route - <	0,73	CompoundMatcher126136
FixCapacity - Capacity - <	SequencedItem - Aircraft - >	0,70	ContextSubsumptionMatcher83729
FixFlow - OccupancyTrafficMonitoringValue - <	SequencedItem - Taxiway - >	0,70	ContextSubsumptionMatcher83808
Flight - Flight - =	Gate - ServiceAtGate - >	0,62	CompoundMatcher18487
Flight - TemporalEnabledEntity - <	Gate - TemporalEnabledEntity - <	0,70	ContextSubsumptionMatcher18808
Flight - WhatIfFlight - >	RadialRoute - Route - <	0,73	CompoundMatcher16336
FlightSpec - FlightConditionCombination - =	AirspaceLayer - Object - <	0,70	ContextSubsumptionMatcher100036
FlightSpec - ObjectWithSchedule - <	Aircraft - Vehicle - <	0,70	ContextSubsumptionMatcher77640
FRDfix - PointReference - <	...		
Gate - Gate - =	CrewMember - PilotInCommand - >	0,70	ContextSubsumptionMatcher25656
Gate - TemporalEnabledEntity - <	AircraftFlow - Flow - <	0,73	CompoundMatcher84259
InternationalAirport - Aerodrome - <	Sector - CircleSector - >	0,73	CompoundMatcher53354
IntervalData - RunwayVisualRange - >	Flight - WhatIfFlight - >	0,70	ContextSubsumptionMatcher34541
METARreportingStation - Aerodrome - >	Flight - TemporalEnabledEntity - <	0,70	ContextSubsumptionMatcher34363
MetCondition - HorizontalVisibility - >	Flight - ReclearanceInFlight - >	0,62	CompoundMatcher34458
MetCondition - WeatherCondition - >	Flight - WhatIfFlight - >	0,62	CompoundMatcher34541
MetCondition - Wind - >	SectorCapacity - Capacity - <	0,73	CompoundMatcher124203
NASfacility - Aerodrome - >	Location - ReferenceLocation - >	0,73	CompoundMatcher45564
...	Location - Object - <	0,71	LexicalSubsumptionMatcher45136
PhysicalRunway - Runway - =	AirspaceInfrastructureComponent - AirspaceLayer - >	0,70	ContextSubsumptionMatcher119861
PhysicalRunway - TemporalEnabledEntity - <	MetCondition - WeatherCondition - >	0,70	ContextSubsumptionMatcher50384
RelativeFix - Object - <	Taxiway - TemporalEnabledEntity - <	0,70	ContextSubsumptionMatcher63643
RelativeFix - PointReference - =	OperationalRunway - Runway - <	0,73	CompoundMatcher25431
	...		
	QRoute - Route - <	0,73	CompoundMatcher128881
	DecomposableSystem - Aircraft - >	0,70	ContextSubsumptionMatcher120329
	AircraftType - Object - <	0,70	ContextSubsumptionMatcher85396
	FixCapacity - Capacity - <	0,73	CompoundMatcher137928

Figure 4.3: Tool support for alignment analysis.

As described in the Introduction in Chapter 1, a matching system is normally composed of a set of individual matchers. Ultimately, the quality of the matching system is based on a combination of the performance of the individual matchers and the strategy used for combining their results. Therefore, a two-step evaluation cycle is required to properly evaluate the impact of a changed technique.

1. Evaluate the local impact of a re-configuration. Here, the alignment

produced by an individual matcher is evaluated against alignments produced by earlier versions of the matcher and the reference alignment in order to verify impact with respect to statistical scoring.

2. Evaluate the global impact of a re-configuration. Here, the consequences of the re-configuration are evaluated at the "global" level, i.e. how the new set of relations and their confidence values produced by the individual matcher affect the aggregation of relations from the ensemble of individual matchers.

4.4 Research Contributions

> "*Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, foundations, and/or methodologies.*"

The results from this thesis contribute by delivering artefacts that can have a positive impact at the business level and by delivering knowledge that can extend the existing knowledge base. From a business-level perspective, the artefacts, as well as ideas generated from their development, have been applied (the BEST project) and are currently applied (the MANU-SQUARE, Digital Water City and ReiseNavet projects) in research projects that aim to solve concrete problems and needs. From a knowledge foundation perspective, the extensions to the state-of-the-art are described in papers and presented at international research conferences (see a full list of papers produced during this work in Appendix A).

One objective of this work has been to assure that the research is conducted in a transparent, reproducible and reliable way. This is important to make a substantial contribution to the application environment as well as to the "knowledge base". This includes making available all source code for all developed artefacts, declaring all dependencies to reused source code and libraries, and ensuring traceability of the research and its evaluation by publishing the following material on GitHub[1].

- All source code related to the different artefacts developed along with documentation (Javadoc).
- A dependency diagram illustrating all external source code libraries and APIs.

[1] https://github.com/audunven

- Material related to the three datasets. This includes the ontologies being matched and the reference alignments holding the correct set of semantic relations.

- Alignments produced by the matching artefacts from this work.

- Alignments as well as related source material from the semantic matching systems BLOOMS [71], S-MATCH [46], STROMA [7], AgreementMakerLight [39] and LogMap [78] that were used in a comparative analysis (see Section 6.1.4).

4.5 Research Rigor

> "*Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artefact, and [...] rigor must be assessed with respect to the applicability and generalizability of the artefact.*"

Before and during the construction of the artefacts, searches were conducted to identify both relevant literature and relevant sources that could support development of the artefacts. Relevant literature was found in general digital libraries, workshop proceedings and other sources of information from events related to the application domain. A synthesis of the literature search is presented in Chapter 3.

Other relevant sources included standards, miscellaneous resources (e.g. WordNet [98]), programming libraries and APIs (e.g. OWL-API [66] and Alignment API [23]), and open source code repositories for existing semantic matching systems. An overview of such external resources is provided in Section 5.7.

The evaluation focused both on the applicability and generalizability of the artefacts. Applicability was determined by comparing the results obtained from the artefacts with other comparable semantic matching systems (see Section 6.1.4). Generalisability was determined by evaluating the artefacts in three different datasets. These datasets represent different application domains and scope, as well as different complexity.

A discussion of the validity, reliability and credibility of the research is presented in Section 7.2.

4.6 Design as a Search Process

> "*Design is essentially a search process to discover an effective solution to a problem.*"

Although the interdependency between the artefacts in this thesis is very strong, the approach followed a staged development process as illustrated in Figure 4.4.

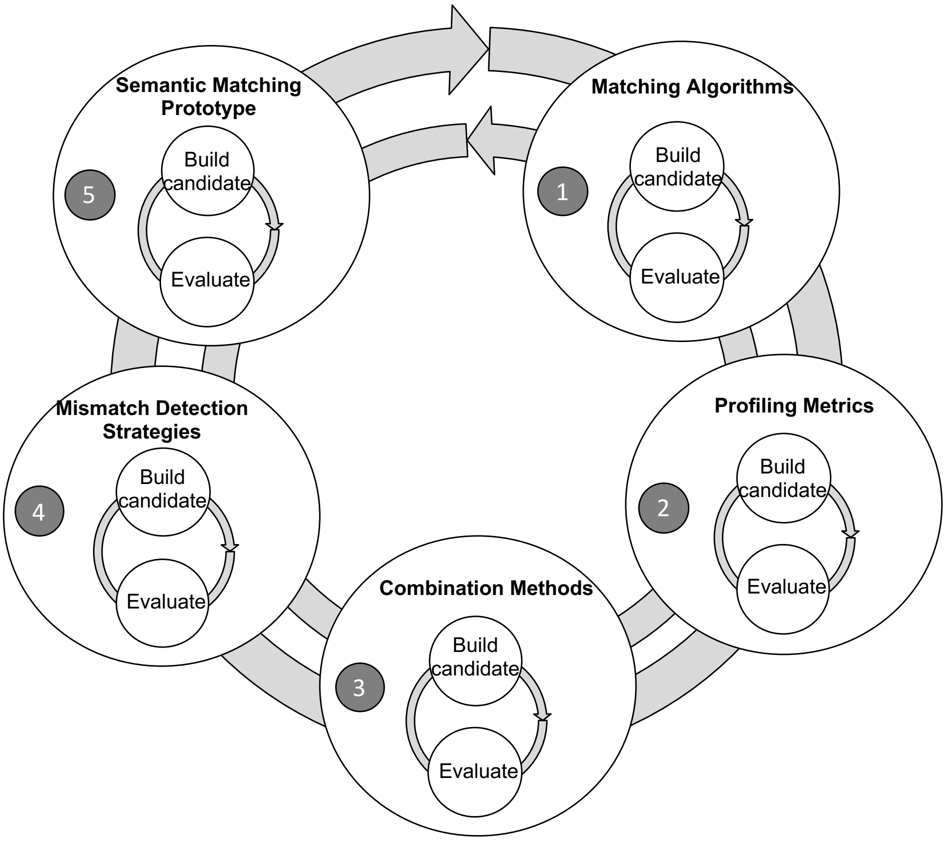


Figure 4.4: Design as a Search Process.

The first stage involved searching for candidate matching algorithms. Supported by solutions and ideas from existing literature, APIs (notably the Alignment API) and open source code repositories candidates that could represent satisfactory solutions (Simon [134] cited in Hevner [65]) were developed and evaluated in "build and evaluate loops". The evaluation benefitted from the benchmark datasets offered by the Ontology Evaluation Alignment Initiative (OAEI)[2]. Initially, for both the subsumption matchers and the equivalence matchers, datasets from the OAEI campaign in 2016[3] were used for the evaluation. These datasets included reference alignments containing only equivalence relations, so reference alignments holding subsumption relations has to be developed by the candidate. In later development phases this dataset was replaced by the dataset from the "Oriented Matching" track from 2011[4].

In stage 2, the focus was on identifying a set profiling metrics that quantitatively describe relevant features from the ontologies to be matched. Existing literature describing related efforts within semantic matching was consulted and new ideas formulated. A broader literature search was conducted within the ontology engineering research area, specifically focusing on literature related to ontology evaluation metrics (e.g. Brank et al. [11] and Tartir et al. [146]). Ideas of candidate metrics were conceived from the specific properties of the matchers. For example, the Compound Ratio (described in Section 5.1.1), which is a measure of the ratio of concept names in the two input ontologies that are compounds, is closely related to the Compound Matcher (described in Section 5.3.1). The Compound Matcher determines a subsumption relation based on whether either of the two concept names are represented as a compound head of the other. A desktop analysis using the matching algorithms from the previous stage was used to test the candidate metrics. The same OAEI datasets as used in stage 1 were used in the evaluation and in addition a second evaluation a dataset containing both equivalence and subsumption relations was developed. This dataset included subsumption and equivalence relations between concepts from the Biblio ontology[5] and the BIBO ontology[6].

Scenarios in the desktop analysis were of the kind:

Statement: The terminological analysis returns a profiling score of 0.22, the structural analysis returns a profiling score of 0.5, while the lexical analysis returns a profiling score of 0.71 (biblio-bibo dataset in Figure 4.5).

Hypothesis: The terminology-based matchers will return alignments with a low F-measure score, the structure-based matchers will return alignments

[2] http://oaei.ontologymatching.org/

[3] http://oaei.ontologymatching.org/2016/

[4] http://oaei.ontologymatching.org/2011/

[5] http://www.cs.toronto.edu/semanticweb/maponto/ontologies/Biblio.owl

[6] http://purl.org/ontology/bibo/

with higher F-measure scores, while the lexical matchers will return alignments with the highest F-measure score.

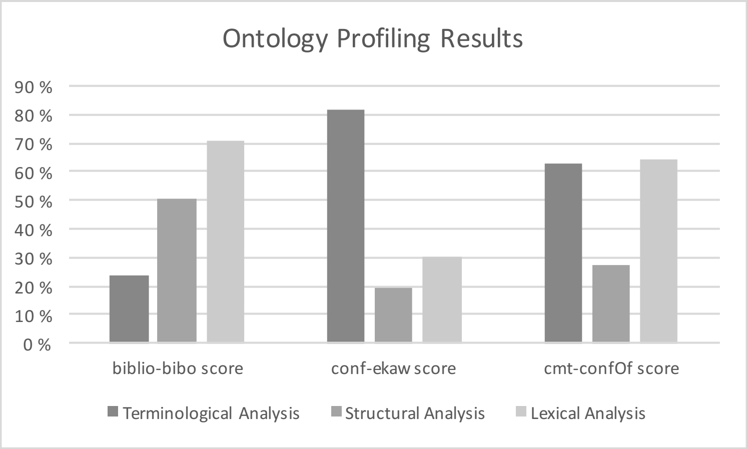


Figure 4.5: Ontology Profiling Results.

With the aim of establishing a good correlation between profiling score and matcher performance (in terms of precision, recall and F-measure of the produced alignment) a number of iterations between matcher development and profiling metric took place in this stage. Moreover, to judge the applicability of the profiling metrics when alignments from individual matchers were combined, some basic combination methods were experimented with at this stage. The evaluation of these methods also called for going back to the "drawing board" and re-designing both matchers and profiling metrics.

Experiences drawn from this stage were reported in Vennesland [151].

In stage 3, the focus was on developing suitable alignment combination methods considering that we had a fairly stable set of matchers and profiling metrics in place from stage 1 and 2. While some papers cover such methods as their core contribution (e.g. CroMatcher and Harmony (see Section 3.4 for a description of both approaches), and some describe quite basic methods (such as the cut method), many papers reveal few details on how the alignment combination is performed. The majority of the relevant papers on this topic describe how to combine equivalence alignments, while how to combine subsumption relations, or equivalence and subsumption relations, is mostly uncovered ground. This stage was the most extensive of all development stages, requiring many revisits to stages 1 and 2 and a large number of candidate methods. The supporting analysis tools described in Section 4.3 were valuable contributions to the evaluation of the many candidate solutions for alignment combination methods. The datasets used in the evaluation described in Chapter 6 were developed during this stage. The development of these datasets is described in Vennesland et al. [152], Vennesland and Aalberg [149], and Gringinger et al. [53].

4.7 Communication of Research

> "*Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.*"

As mentioned in Section 4.4, all source code and evaluation material for all artefacts developed in this thesis are made available on-line on GitHub. This should enable technology-oriented audiences to both understand the processes by which the artefacts were both constructed and evaluated. With respect to management-oriented audiences, a different kind of presentation is needed. Explaining the utility of the artefacts in terms of how they impact on organizational roles and processes, as well as the concrete benefits such as increased revenue, cost savings, and increased efficiency are more convincing arguments here.

Part III

Implementation and Evaluation

5 Development of Semantic Matching Artefacts

This chapter describes the artefacts that have been developed in this work. These include equivalence- and subsumption matchers, ontology profiling techniques, mismatch detection strategies and alignment combination techniques.

Figure 5.1 illustrates the entire matching process. The process begins by profiling the ontologies to be matched according to the metrics described in Section 5.1. The results from the ontology profiling process are used both to select the most appropriate set of individual matchers and for dynamically configuring the confidence value each matcher assigns its identified semantic relations. This matcher selection and configuration is described in Section 5.4. Once the individual matchers have been selected and configured, the Matcher Execution consists of running the selected equivalence and subsumption matchers in parallel. The equivalence matchers are described in detail in Section 5.2 and the subsumption matchers are described in Section 5.3. A number of mismatches contributed by differing conceptualisation and explication factors (see Section 2.3) can significantly reduce the quality of ontology alignment. In order to mitigate the negative effect of ontology mismatches, two mismatch detection techniques have been developed in this work. These techniques try to identify mismatches in the equivalence alignments and filter them out of the produced alignment. The mismatch detection techniques are described in Section 5.5. Finally, in order to produce a final alignment that is returned to the user, the alignments produced by the individual equivalence and subsumption matchers are combined in the alignment combination process. This process includes different strategies for aggregating an optimal set of relations from the individual alignments. The alignment combination methods experimented with in this work are presented in Section 5.6.

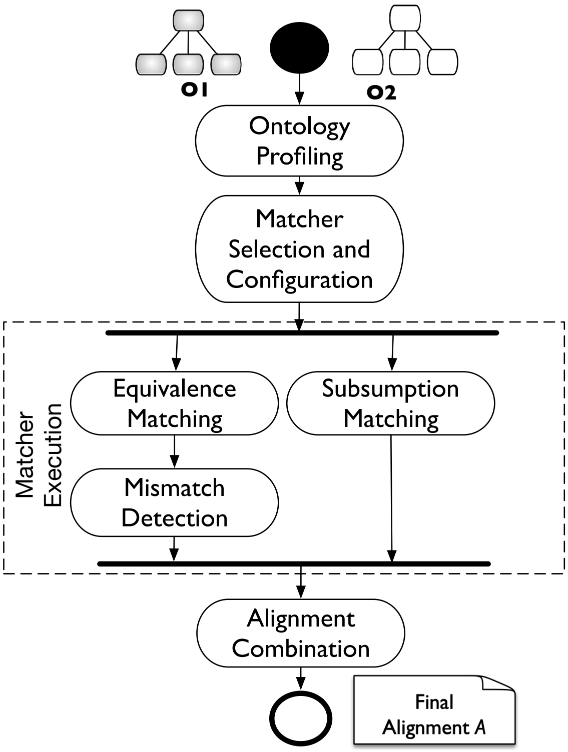


Figure 5.1: An overview of the semantic matching process.

Finally, in Section 5.7 the most significant external sources and software libraries used in the development of the artefacts are described.

5.1 Ontology Profiling

First, after the input ontologies have been pre-processed and parsed to an appropriate representation, a set of metrics that characterise the input ontologies are computed in the ontology profiling. These metrics characterise the terminological, structural and lexical profile of the input ontologies and are computed as an average metric for both ontologies. The ontology profiling metrics are used in two separate processes:

1. To determine whether or not a certain matcher should be included in the matcher ensemble used to match the input ontology. This process

is called *Matcher Selection.*

2. As a weight parameter that together with the initial similarity score determined by the matcher is used for computing a confidence value for each semantic relation in the alignment returned by the matcher. This task is called *Matcher Configuration.*

These two processes are described more in detail in Section 5.4. The metrics included in the ontology profiling step are described in the following sections.

5.1.1 Terminological Analysis

There are three metrics included in the terminological analysis. These are named *Compound Fraction*, *Corpus Coverage* and *Definition Coverage.*

The Compound Fraction metric is based on an analysis of whether a concept name represents a compound or not. A compound is a word that consists of at least two constituent words, such as *PhDThesis.* Here, the head of the compound is *Thesis*, while *PhD* is a modifier that further specifies the type of thesis. Compound Fraction is computed using the number of compound class names as numerator and the number of classes in both input ontologies as denominator. If the Compound Fraction, that is, the representation of compounds, is high, this suggests that a matcher capable of exploiting such linguistic structures should be included in the ensemble of matchers.

The Corpus Coverage metric analyses how many individual tokens from the two input ontologies reside in a corpus representing word embeddings. The set of individual tokens are extracted from class names, labels, and natural language definitions of the two input ontologies. The corpus representing the word embeddings is a file that includes a word and associated vectors on each line. The Corpus Coverage metric is computed as the fraction of ontology tokens extracted from the input ontologies that are represented as words in the word embedding file over all ontology tokens extracted from the input ontologies.

The Definition Coverage metric aims to capture how well defined the concepts in the input ontologies are. It is calculated by measuring the fraction of concepts that have a natural language definition in each of the two ontologies, and the minimum of the two fractions is used to define the Definition Coverage. The reason for this is that even if one of the ontologies have no definitions, while the second one has all concepts annotated the Definition Coverage measure would be fairly high at 0.5 if an average across both ontologies was applied. As some ontologies may include very short definitions, not revealing much of the semantic intention of the concept being annotated, a minimum of 10 words (after stopword removal) is required.

5.1.2 Structural Analysis

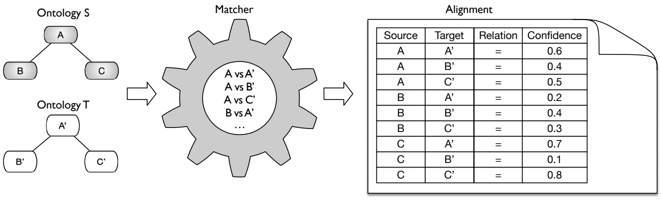
In order to capture the structural characteristics of the input ontologies two metrics are used. The *Property Fraction* measures the extent to which the input ontologies include properties (data- and object properties). The Property Fraction is computed as the fraction of classes that are associated with data- or object properties over the total number of classes in the two ontologies. The other metric in the structural analysis, *Structural Profile*, considers the coverage of subclasses and superclasses in the two ontologies, and is computed as the fraction of classes that have sub- or superclasses associated with them over all classes in both ontologies.

5.1.3 Lexical Analysis

The lexical analysis consists of a single metric called *Lexical Coverage*. Lexical Coverage measures the percentage of class names present in the WordNet lexicon [98]. If the class name happens to be a compound, we break the compound down to its compound parts, and consider a match if any of its parts reside in these lexicons. A similar metric (WordNet Coverage) was used in Cruz et al. [21], however then disregarding the compound parts of a word.

5.2 Equivalence Matchers

This section describes matchers developed for computing equivalence relations between ontology concepts. All equivalence matchers perform a pairwise similarity computation of all concepts from the respective input ontologies. The result from each matcher's computation is an alignment holding a set of equivalence relations with an associated confidence value determined by the matcher. This is illustrated in Figure 5.2.



Source	Target	Relation	Confidence
A	A'	=	0.6
A	B'	=	0.4
A	C'	=	0.5
B	A'	=	0.2
B	B'	=	0.4
B	C'	=	0.3
C	A'	=	0.7
C	B'	=	0.1
C	C'	=	0.8

Figure 5.2: Input and output from matchers.

Several of these matchers either re-use or extend techniques developed by others. Table 5.1 provides an overview of all implemented matchers along with a description of knowledge re-use and novelty.

Table 5.1: Summary of equivalence matchers

Matcher	Description
Word Embedding Matcher	Others have applied Word Embedding as a means for matching ontologies [119, 161, 77, 117], however the approach of extracting and processing embedding vectors for compound concept names and combine these individual vectors into a single *name vector* is to the best of our knowledge novel.
Definitions Equivalence Matcher	This matcher averages the embedding vectors associated with the concept name and embedding vectors associated with other natural language descriptions (i.e. annotations in the form of rdfs comments) tied to the ontology concepts.
Property Equivalence Matcher	This matcher employs the Core Concept technique from Cheatham et al. [18] in order to match properties, but extends it by performing a more comprehensive lexical analysis afterwards.
Lexical Equivalence Matcher	Using synonym sets retrieved from WordNet is a well-known technique for computing equivalence and is for example described in [89]. However, we are not aware of earlier works that conduct the same compound analysis nor the same set similarity technique.
Graph Equivalence Matcher	This matcher is inspired by the Structural Proximity method proposed by Hu et al. [69]. However, an adapted variant based on the average distance to common superclasses is used by this matcher (see a description in Section 3.1.2). Furthermore, this matcher uses the ISub string matching algorithm [139] instead of string equality to determine similarity between parent nodes.

5.2.1 Word Embedding Equivalence Matcher

We start by explaining how the word embedding approach is used for the two matchers Word Embedding Matcher and the Definitions Equivalence Matcher. Using Word2Vec and the Skip-Gram model (described in Section 2.4), a word-to-vector representation has been obtained from two large natural text corpora, namely Wikipedia and Skybrary. Wikipedia represents general knowledge irrespective of application domain, whereas Skybrary contains knowledge targeted for the air traffic domain. Section 5.7.1 describes additional details on how these two corpora have been processed.

As a preparation for the matching, the word-to-vector representation for each corpus is represented as a hashmap where each word in the corpus is mapped to its vectors generated from the Word2Vec process. During each matching operation each ontology to be matched is represented as a separate word-to-vectors hashmap where all concept names and words in annotations (i.e. rdfs:comments) are represented as key and the associated vectors are represented as value (see Figure 5.3). In order to match two ontology concepts the Word Embedding Equivalence Matcher retrieves a vector representation of their class name, and computes similarity using the Cosine metric.

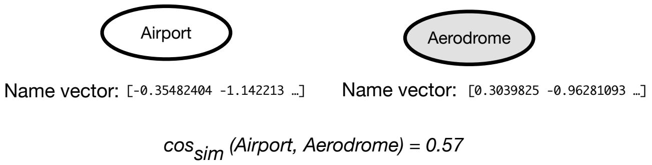

Airport
Aerodrome
Name vector: [-0.35482404 -1.142213 …]
Name vector: [0.3039825 -0.96281093 …]
cos_sim (Airport, Aerodrome) = 0.57

Figure 5.3: The Word Embedding Matcher computes similarity from name vectors.

There are three types of vector representations:

- **Name vector**: A name vector is represented by the extracted set of vectors associated with the concept name of an ontology concept. Very often concept names are represented as compounds. If that is the case, we decompose the compound into its parts, retrieve the vectors for each part and average the vector components into a single vector representation.
- **Comment vector**: A comment vector is established from the set of vectors for each token associated with the natural language description (i.e. annotations in the form of rdfs:comments) related to a concept. The comment vector is pre-processed by removing stopwords from the annotations. We then extract vectors related to each remaining individual word (token) represented in the annotation and average all of these into a single vector representation for each annotation (comment).
- **Global vector**: A global vector represents the averaged vector from a name vector and a comment vector.

The Word Embedding Equivalence Matcher only considers the name vector and computes a similarity score based on that.

The Word Embedding Equivalence Matcher operates as described in Algorithm 1.

Algorithm 1 Word Embedding Matcher

```
Input: Vector hashmap $VHM$, source ontology $O_s$, target ontology $O_t$
Output: Alignment produced by Word Embedding Matcher $A_{WEM}$
 1: function $computeWEMAlignment(VHM, O_s, O_t)$
 2: $S_{name}$, $T_{name}$, $SVHM$, $TVHM$, $A_{WEM} \leftarrow \emptyset$
 3: $WEM_{sim} \leftarrow 0$
 4: for all $c_s \in O_s$ do
 5:     $S_{name} \leftarrow getNameVectors(c_s, VHM)$
 6:     if $S_{name} \neq \emptyset$ then
 7:         $SVHM \leftarrow SVHM \cup addNameVectors(c_s, S_{name})$
 8:     end if
 9: end for
10: for all $c_t \in O_t$ do
11:     $T_{name} \leftarrow getNameVectors(c_t, VHM)$
12:     if $T_{name} \neq \emptyset$ then
13:         $TVHM \leftarrow TVHM \cup addNameVectors(c_t, T_{name})$
14:     end if
15: end for
16: for all $c_s \in O_s$ do
17:     for all $c_t \in O_t$ do
18:         if $contains(SVHM, c_s)$ and $contains(TVHM, c_t)$ then
19:             $WEM_{sim} \leftarrow computeCosSim(getVectors(SVHM, c_s), getVectors(TVHM, c_t))$
20:         else
21:             $WEM_{sim} \leftarrow 0$
22:         end if
23:         $A_{WEM} \leftarrow A_{WEM} \cup addRelation(c_s, c_t, =, WEM_{sim})$
24:     end for
25: end for
26: return $A_{WEM}$
27: end function
```

In lines 4-15 word-to-vector hashmaps ($SVHM$ and $TVHM$) are constructed for the two ontologies to be matched (O_s and O_t). These hashmaps contain a key representing each concept (represented by its concept name) and the name vector extracted from the vector hashmap as described in the following. We extract the name vector for each concept c_s from ontology O_s on line 5. The $getNameVectors$ method extracts the name vector associated with concept c_s from the vector hashmap VHM. The same operation is performed for concept c_t on line 11. The pairwise matching of the concepts in the two ontologies begins on line 16. On line 18 we check if both concepts being matched are represented in the respective word-to-vector hashmap. If so, we extract the vectors associated with these concepts and compute a similarity score using the cosine measure on line 19. If either of the concepts are not represented in the respective hashmap, the similarity is set to 0 (line 21). Once the similarity computation has finished, the relation (c_s and c_t) is added to the alignment A_{WEM} along with the similarity score computed for this relation on line 23. The method returns a complete alignment A_{WEM} on line 26.

5.2.2 Definitions Equivalence Matcher

The Definitions Equivalence Matcher identifies equivalent concepts by using the cosine similarity between global vectors, see Figure 5.4. However, if a concept being matched does not have a definition in terms of an rdfs:comment (i.e. there is no comment vector), we use the name vector to represent it during the matching operation.

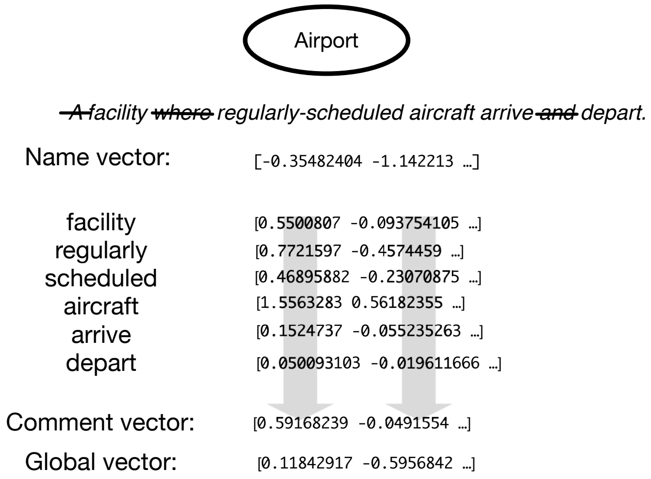

Airport
~~A~~ facility ~~where~~ regularly-scheduled aircraft arrive ~~and~~ depart.
Name vector: [-0.35482404 -1.142213 …]
facility [0.5500807 -0.093754105 …]
regularly [0.7721597 -0.4574459 …]
scheduled [0.46895882 -0.23070875 …]
aircraft [1.5563283 0.56182355 …]
arrive [0.1524737 -0.055235263 …]
depart [0.050093103 -0.019611666 …]
Comment vector: [0.59168239 -0.0491554 …]
Global vector: [0.11842917 -0.5956842 …]

Figure 5.4: Each ontology concept is described by name vectors and global vectors.

Algorithm 2 shows how the Definitions Equivalence Matcher produces an alignment. This approach is very similar to the Word Embedding Equivalence Matcher approach described in Section 5.2.1, except for that instead of name vectors global vectors are used by Definitions Equivalence Matcher (indicated in red in the algorithm).

In lines 4-15 word-to-vector hashmaps ($SVHM$ and $TVHM$) are constructed for the two ontologies to be matched (O_s and O_t). These hashmaps contain a key representing each concept (represented by its concept name) and the value for each key is represented by the corresponding global vector. We extract the global vector for each concept c_s from ontology O_s on line 5. The $getGlobalVectors$ method extracts the name vector and the comment vector for each token in the definition (i.e. rdfs:comment) associated with

Algorithm 2 Definitions Equivalence Matcher

```
Input: Vector hashmap $VHM$, source ontology $O_s$, target ontology $O_t$
Output: Alignment produced by Definitions Equivalence Matcher $A_{DEM}$
 1: function $computeDEMAlignment(VHM, O_s, O_t)$
 2: $S_{global}$, $T_{global}$, $SVHM$, $TVHM$, $A_{DEM} \leftarrow \emptyset$
 3: $DEM_{sim} \leftarrow 0$
 4: for all $c_s \in O_s$ do
 5:     $S_{global} \leftarrow getGlobalVectors(c_s, getDefinition(c_s), VHM)$
 6:     if $S_{global} \neq \emptyset$ then
 7:         $SVHM \leftarrow SVHM \cup addGlobalVectors(c_s, S_{global})$
 8:     end if
 9: end for
10: for all $c_t \in O_t$ do
11:     $T_{global} \leftarrow getGlobalVectors(c_t, getDefinition(c_t), VHM)$
12:     if $T_{global} \neq \emptyset$ then
13:         $TVHM \leftarrow TVHM \cup addGlobalVectors(c_t, T_{global})$
14:     end if
15: end for
16: for all $c_s \in O_s$ do
17:     for all $c_t \in O_t$ do
18:         if $contains(SVHM, c_s)$ and $contains(TVHM, c_t)$ then
19:             $DEM_{sim} \leftarrow computeCosSim(getVectors(SVHM, c_s), getVectors(TVHM, c_t))$
20:         else
21:             $DEM_{sim} \leftarrow 0$
22:         end if
23:         $A_{DEM} \leftarrow A_{DEM} \cup addRelation(c_s, c_t, =, DEM_{sim})$
24:     end for
25: end for
26: return $A_{DEM}$
27: end function
```

concept c_s from the vector hashmap VHM and creates a global vector that is set to the S_{global} variable. The same operation is performed for concept c_t on line 11. The pairwise matching of the concepts in the two ontologies begins on line 16. On line 18 we check if both concepts being matched are represented in the respective word-to-vector hashmap. If so, we extract the vectors associated with these concepts and compute a similarity score using the cosine measure on line 19. If either of the concepts are not represented in the respective hashmap, the similarity is set to 0 (line 21). Once the similarity computation has finished, the relation (c_s and c_t) is added to the alignment A_{DEM} along with the similarity score computed for this relation on line 23. The method returns a complete alignment A_{DEM} on line 26.

5.2.3 Property Equivalence Matcher

The Property Equivalence Matcher measures the similarity of the properties associated with the concepts to be matched and uses that to infer concept similarity. Both object properties and data properties where the concepts to be matched represent the domain or range class are collected into single sets

$C_{x^{prop}}$ and $C_{y^{prop}}$ and compared with Jaccard. However, property names are challenging to process for a number of reasons. First of all, there is a large variety of conventions used when naming a property. For example, should a prefix be added to the property name to distinguish its intended meaning, for example -is or -has, or should the property name consist of a single word aiming to capture the relationship defined by the property? Another challenge is that property names are often compounds with multiple parts, and it can be difficult to determine which of its parts to use to capture the semantic essence of the property.

In order to match properties the Property Equivalence Matcher tries to identify the *core concept* of each property, inspired by the works of Cheatham et al. [18], which focuses on equivalence matching of properties. The core concept is either the first verb in the label that is greater than 4 characters or, if no such verb exists, the first noun in the label, together with any adjectives that qualify that noun. A Part-of-Speech (POS) tagger is used for differentiating verbs, nouns and adjectives in a property name. Currently, the POS tagger from the Stanford CoreNLP API[1] is used.

The Property Equivalence Matcher extends the approach from Cheatham as follows:

- Once the core concepts are extracted, potential synonyms of them are retrieved from WordNet. Both nouns and verbs are retrieved. As the example illustrated in Figure 5.5 shows, the object property between *Thesis* and *Person* is *isAuthorOf*, where the core concept is 'author'. In the other ontology, the object property *writer* links *Dissertation* to *HumanAgent*, and the core concept is 'writer'. The synonyms retrieved for'author' are 'writer', 'poet' and 'correspondent', 'authoress' and 'co-author' whereas for 'writer' synonyms are 'author', 'poet', and 'diarist' and 'essayist'.

- The similarity of the two respective synonym sets is calculated using Jaccard, but with a change in that the similarity score is increased if the respective sets include the core concept of the opposite property. Take the example in Figure 5.5. Here, the set of synonyms for 'author' include 'writer' and the set of synonyms for 'writer' includes 'author' as one of the synonyms. This should be "rewarded" somehow as it strengthens the belief that these properties have a strong similarity. This reward is accomplished by moving them out of the union set and

[1] https://stanfordnlp.github.io/CoreNLP/api.html

into the intersection set. The result of this is that a score of 0.33 ($\frac{3}{9}$) is returned instead of a score of 0.11 ($\frac{1}{9}$). In case the nominator (i.e. the cardinality of the intersection set) is greater than the denominator (i.e. the cardinality of the union set), we set an upper bound of 1, so the returned score is in the range [0..1]. Furthermore, in order to have some flexibility with respect to naming differences we do not use exact string equality, but the ISub string matcher (see Section 2.5) with a similarity threshold of 0.7. This threshold was determined experimentally.

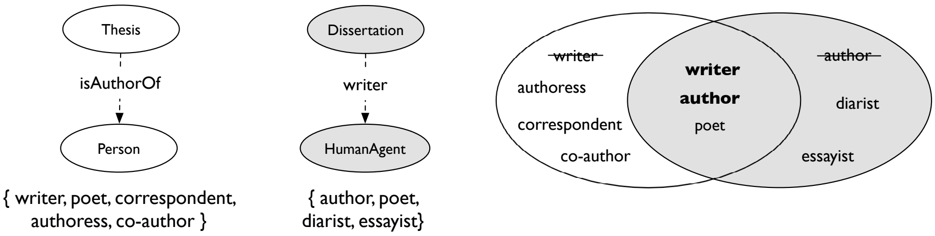


Figure 5.5: The Property Equivalence Matcher identifies equivalence relations using the notion of a core concept combined with relaxed synonym similarity

5.2.4 Lexical Equivalence Matcher

The Lexical Equivalence Matcher uses WordNet as a lexical resource for computing equivalence relations between ontology concepts. It is well-known that the use of WordNet in ontology matching is a double-edged sword [36]. While it can be a very valuable resource and capture relations that other techniques would miss, it can also reduce the overall quality of a matching process due to its low coverage of domain specific concepts as well as a sparse amount of compound words. In this matcher, we try to tackle the shortcomings of WordNet by using an approach that combines de-compounding with the use of a semantic similarity measure and an analysis of synonyms associated with de-compounded parts from the concepts to be matched. The equivalence relations computed by this matcher is based on a combination of the semantic similarity measure Jiang-Conrath [75] and Jaccard [70] set similarity (see Section 2.5.1 for a description of both Jiang-Conrath and Jaccard).

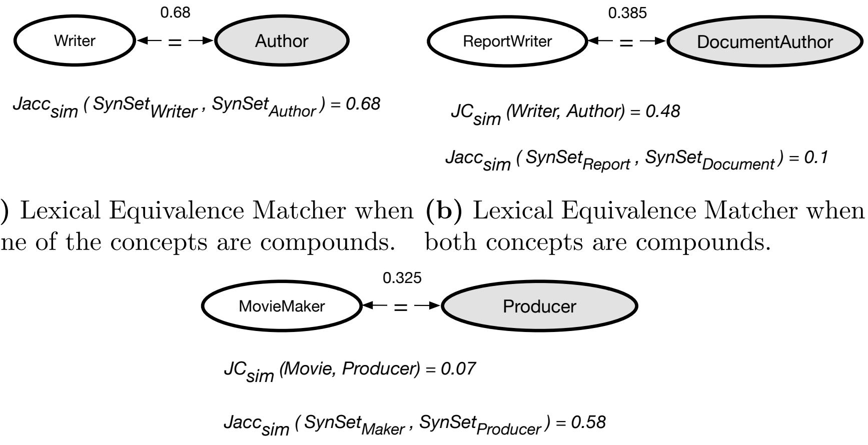
The procedure performed by the Lexical Equivalence Matcher is as follows:

1. If none of the concepts are compounds, we retrieve the sets of WordNet synonyms for both concepts and measure their similarity using Jaccard

similarity of sets. This is illustrated in Figure 5.6(a).

2. If both concepts are compounds, we split them and separate their compound heads and compound modifiers. The similarity score between these two concepts is a composite score. For the compound heads we measure their similarity using Jiang-Conrath as we want to assure that they have some semantic relatedness in the first place. For the compound modifiers we retrieve their WordNet synonyms and measure their similarity using Jaccard. If a compound modifier is a compound itself, we split its constituent parts, retrieve synonyms for each part into a joint set of synonyms for each concept, and compute a similarity score using Jaccard. The final similarity score is based on weighting the Jiang-Conrath score with 75 % and the Jaccard score with 25 %, basically giving more priority to similar compound heads than similar modifiers. However, if only the compound heads are similar, but there are no WordNet synonyms for the respective compound modifier(s), we return a score of 0 since similarity between the compound heads is not sufficient grounds for saying that the two concepts are similar. This sub-procedure is illustrated in Figure 5.6(b).

3. If only one of the concepts is a compound while the other concept is an "atomic" word, it is considered less likely that the two concepts form an equivalence relation. However there are situations where such a pattern may occur, for example *WeatherPerson* = *Meteorologist* or *ComicStrip* = *Cartoon*. In such situations, the Lexical Equivalence Matcher returns a similarity score by computing the Jiang-Conrath score between the compound modifier(s) of the compound concept and the full name of the other concept, and the Jaccard set similarity between the synonyms of the compound head of the compound concept and synonyms of the full name of the other concept. These two scores are evenly weighted in the final score. We have included two clauses here though: (1) If the compound head of the compound concept equals the full name of the other concept (e.g. MasterThesis vs. Thesis), we return a score of 0 since in such a case it is more likely a subsumption relation than an equivalence relation. (2) If either of the compound modifiers of the compound concept equals the full name of the other concept (e.g. BookPage vs. Book) we return a score of 0 because in this case it is more likely a meronymic relation than an equivalence relation. This sub-procedure is illustrated in Figure 5.6(c).


0.68
Writer
=
Author
Jacc_sim (SynSet_Writer , SynSet_Author) = 0.68
0.385
ReportWriter
=
DocumentAuthor
JC_sim (Writer, Author) = 0.48
Jacc_sim (SynSet_Report , SynSet_Document) = 0.1
0.325
MovieMaker
=
Producer
JC_sim (Movie, Producer) = 0.07
Jacc_sim (SynSet_Maker , SynSet_Producer) = 0.58

(a) Lexical Equivalence Matcher when none of the concepts are compounds.

(b) Lexical Equivalence Matcher when both concepts are compounds.

(c) Lexical Equivalence Matcher when one of the concepts is a compound.

Figure 5.6: Lexical Equivalence Matcher.

5.2.5 Graph Equivalence Matcher

An ontology is a labelled Rooted Directed Acyclic Graph (RDAG), with a single root (thing), concepts represent nodes and the edges between the nodes are relationships (either is-a or object properties). Such a graph representation enables the utilisation of algorithms that exploit the graph structure in order to compute similarity between concepts from different ontologies.

Inspired by the Wu-Palmer algorithm (described in Section 2.5.2) this matcher identifies the structural proximity of two nodes using the following steps:

1. Calculate the distance (number of edges) from the two nodes $n1$ and $n2$ (ontology concepts to be matched) to their root (thing) as $n1_{dist}$ and $n2_{dist}$ respectively.

2. Identify the set of ancestor nodes to n_1 and n_2 with similarity above a certain threshold.

3. Calculate the distance from each pair of ancestor nodes to the respective graph's root and calculate the average distance $avgAnc_{dist}$.

Then, when the above distances have been retrieved, compute the equivalence score between two nodes as follows:

$$GraphSim(n_1, n_2) = \frac{(2 * avgAnc_{dist})}{(n1_{dist} + n2_{dist})} \tag{5.1}$$

The rationale of this matcher is to factor in the collective (lexical) similarity of the parent nodes of n_1 and n_2 and use that to approximate similarity between the two nodes (concepts) being matched. Different from the Wu-Palmer algorithm, the matcher does not consider the lowest common subsumer (LCS), but rather the average distance of parent nodes that have a lexical similarity above a given threshold.

Figure 5.7 illustrates how the Graph Equivalence Matcher operates.

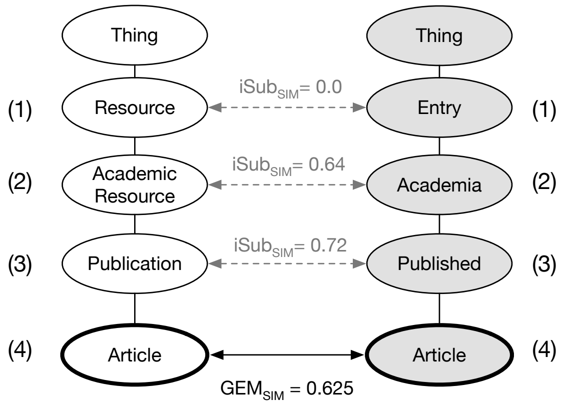


Figure 5.7: The Graph Equivalence Matcher computes a similarity score by taking into account the structural proximity of two concepts

Here, the two concepts (nodes) to be matched are *Article* in ontology 1 and *Article* in ontology 2. The distance from both nodes to the root is 4 (as indicated by numbers in parenthesis), so $n1_{dist}$ and $n2_{dist}$ are both 4. The ISub similarity (see Section 2.5.1) between the ancestors *Publication* and *Published* is 0.72, at the level above the similarity between *AcademicResource* and *Academia* is 0.64, while the similarity is 0.0 for ancestors *Resource* and *Entry*. From these computations we consider the ancestor pairs {*Publication*, *Published*} and {*AcademicResource*, *Academia*} similar using a similarity threshold of 0.6. Next, the matcher computes an average distance $avgAnc_{dist}$ considering the distance of each ancestor node in these pairs up to the root node (Thing). The average distance from the similar ancestors to the root is 2.5. From this, the computation of the Graph Similarity is:

$$GraphSim(Article, Article) = \frac{(2 * 2.5)}{(4 + 4)} = 0.625 \tag{5.2}$$

5.3 Subsumption Matchers

Four subsumption matchers have been developed in this work. Table 5.2 describes how their development is inspired from previous work and what are new contributions to semantic matching.

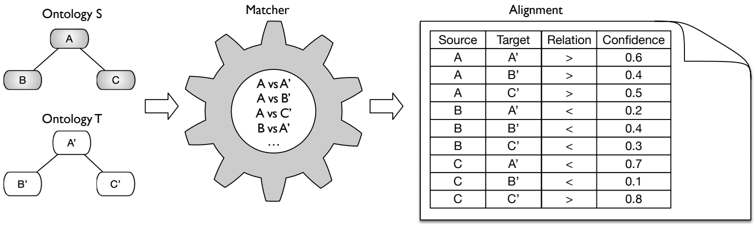
Table 5.2: Summary of subsumption Matchers

Matcher	Description
Compound Matcher	The Compound Matcher is inspired by the Compound Strategy described in Arnold and Rahm [7]. Their strategy is extended in that the Compound Matcher also considers synonyms of the compound head retrieved from WordNet.
Context Subsumption Matcher	The surrounding class structure of two concepts to be matched is also used by Arnold and Rahm [7]. However, from the approach described in their paper, it seems they only consider the superclasses of two ontology concepts when determining a subsumption relation, not their subclasses.
Lexical Subsumption Matcher	The Lexical Subsumption Matcher uses an approach that combines two techniques in order to identify a subsumption relation. The first technique, where WordNet hyponyms are used to indicate subsumption relationship is also used in Arnold and Rahm [7]. The other technique, which is used to sharpen the precision of the hyponym set, uses the Resnik similarity metric [121] to qualify the subsumption relation by considering the information content of the source and target concepts.
Definitions Subsumption Matcher	The Definitions Subsumption Matcher uses Lexico-Syntactic Patterns [62] to detect subsumption, and a similar approach performed by others has not been identified.

As with the equivalence matchers presented in Section 5.2, the subsumption matchers perform a pairwise computation of all concepts from the respective input ontologies in order to derive semantic relatedness. The result from the computation is an alignment holding a set of subsumption relations with an associated confidence value determined by the matcher. This is illustrated in Figure 5.8.

5.3.1 Compound Matcher

The Compound Matcher identifies subsumption relations between entities reusing principles from the compound strategy from Arnold and Rahm [7] and the compound noun analysis used by Cruz et al. [22]. Here, compounds in entity names are identified and employed as an indicator of a subsump-



Source	Target	Relation	Confidence
A	A'	>	0.6
A	B'	>	0.4
A	C'	>	0.5
B	A'	<	0.2
B	B'	<	0.4
B	C'	<	0.3
C	A'	<	0.7
C	B'	<	0.1
C	C'	>	0.8

Figure 5.8: Input and output from matchers.

tion relation (e.g. *Research − Project* is subsumed by *Project*). Different strategies for detecting compound parts were experimented with during the development of this matcher, for example using a list of all English nouns as a basis for detecting if several of these were included in a single concept name. This strategy proved very error-prone as many of the words in the list were accidentally considered words when being simply substrings, resulting in that may concept names that were not compounds were considered as such. For example, *Content*, where both *ten* and *tent* are listed as separate nouns in the word lists. De-compounding, as this process is often termed, is a very challenging task, and open sourced and efficient compound splitting software for the English language is difficult to come by. However, for the purposes of this matcher a quite naive implementation worked pretty well. Here, the compound parts are identified using a regular expression as follows:

```
( ? < ! ( ˆ | [A–Z] ) ) ( ? = [A–Z] ) | ( ? < ! ˆ ) ( ? = [A–Z] [ a–z ] )
```

This regular expression extracts all parts of a string that begins with an upper-cased letter and works well for camel-cased and Pascal-cased concept names, both notations that are typically used for concept names in ontologies. This means that for the concept name *AcademicResearchProject* the following set of constituent compound parts is extracted: {'academic', 'research', 'project'}.

Note that a concept with fewer compound modifiers is considered a stronger indication of a direct subsumption relation than a concept with more compound modifiers, and this is acknowledged by the scoring function of this matcher (see below). The rationale is that it is likely that *Research − Project* is an intermediary concept in between *AcademicResearchProject* and *Project*.

We extend the compound strategy by also considering synonyms to the head of a compound word. This is illustrated in Figure 5.9. With this extension, the concept *ResearchProject* can be considered a subclass of the concept *Project*, but also *Undertaking* (where *Undertaking* is synonymous with *Project*) according to WordNet, and similarly, *Research-Undertaking* can be considered a subclass of *Project*.

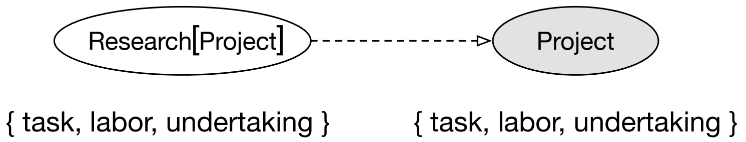


Figure 5.9: Compound Matcher

The scoring function of the Compound Matcher is as follows:

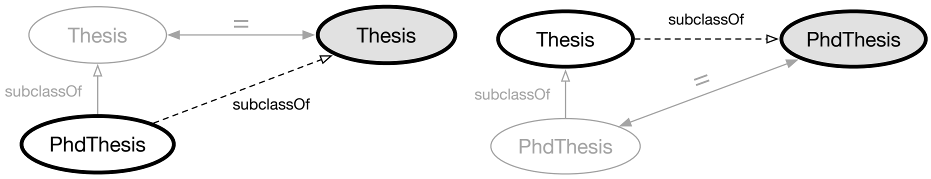
- If the source concept is a compound, its compound head equals the full name of the target concept and the source concept consists of only one compound modifier: source < target and the confidence value is 1.0, or vice versa.
- If the source concept is a compound, its compound head equals the full name of the target concept and the source concept consists of two compound modifiers: source < target and the confidence value is 0.75, or vice versa.
- If the source concept is a compound, its compound head equals the full name of the target concept and the source concept consists of three or more compound modifiers: source < target and the confidence value is 0.50, or vice versa.
- If the source concept is a compound, a synonym of its compound head equals the full name of the target concept and the source concept consists of one compound modifier: source < target and the confidence value is 0.75, or vice versa.
- If the source concept is a compound, a synonym of its compound head equals the full name of the target concept and the source concept consists of two compound modifiers: source < target and the confidence value is 0.50, or vice versa.
- If the source concept is a compound, a synonym of its compound head equals the full name of the target concept and the source concept consists of three or more compound modifiers: source < target and the confidence value is 0.25, or vice versa.

5.3.2 Context Subsumption Matcher

The Context Subsumption Matcher identifies a subsumption relation between two concepts based on their context, i.e. their parent classes and child classes.

An example that illustrates the approach taken by the Context Subsumption Matcher is provided in Figure 5.10. Here, in the example illustrated in Figure 5.10(a), the Context Subsumption Matcher concludes that the source concept *PhdThesis* (white) should be a subclass of the target concept *Thesis* in ontology 2 (grey), since this latter class is equivalent with the *Thesis* concept in ontology 1 (white). Alternatively, as exemplified in Figure 5.10(b), the Context Subsumption Matcher derives that the target concept *PhdThesis* (grey) should be subsumed by the source concept *Thesis* (white) since the latter has a subclass that is equal to the target concept.

A similar approach is also used by the STROMA matching system [7], which is described in Section 3.2. In STROMA this is called the *Structure Strategy*.



(a) Deriving a subsumption relation from a parent's equivalence relation to a target concept.

(b) Deriving a subsumption relation from a parent's equivalence relation to a target concept.

Figure 5.10: Context Subsumption Matcher.

5.3.3 Lexical Subsumption Matcher

This matcher is based on the combination of two features: hyponyms in WordNet and information content (see Section 2.5.3). The first feature is the dominant feature of the two, and is represented by hyponyms retrieved from WordNet. If the term used to express the source concept is present in the hyponyms of the term used to express the target concept, the source concept is subsumed by the target concept. This approach is similar to the *Background Knowledge Strategy* of Arnold and Rahm [7] and it is illustrated in Figure 5.11 where the concept *Article* matches one of the hyponyms associated with *Publication*.

The second feature is a qualifying feature that aims to filter out falsely determined hyponym relations between two concepts on the basis of their information content. As described in Section 5.2.4, using WordNet can often lead to a large number of false positives. Hence, we want to strengthen the belief that two concepts are in fact semantically related using the Resnik [121] semantic similarity measure.

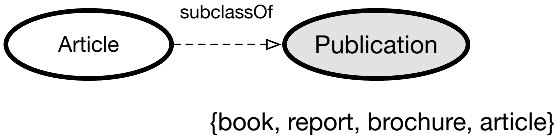


Figure 5.11: Lexical Subsumption Matcher

The similarity score of this matcher is determined by the following rule: If a source concept C_s is present in the hyponym set of a target concept C_t and the Resnik similarity score is above a defined threshold T (in our experiments we have used 0.75): source $<$ target with a confidence of T; and vice versa.

5.3.4 Definitions Subsumption Matcher

The Definitions Subsumption Matcher combines *lexico-syntactic patterns* and *lexical processing* of definitions associated with ontology concepts. This approach is based on Hearst's work on automatically identifying hyponymy relations from unstructured texts [62].

First, the natural language definitions (i.e. rdfs:comments) for the two concepts to be matched are extracted if they contain any of the defined lexico-syntactic patterns. These definitions are pre-processed by removing stopwords and other non-alphabetic characters. Then the definitions are tokenized and each token is lemmatized using the Stanford SimpleNLP API[2]. Next, using the patterns described in Table 5.3, the matcher infers hyponym or hypernym relations between two concepts C_s and C_t as follows:

If the natural language definition of a source concept C_s contains either of these patterns and the name of the target concept C_t equals a noun that follows a pattern the matcher concludes that there is a subsumption relation source $>$ target between these two concepts (and vice versa).

A challenge with these lexico-syntactic pattern is that although they can automatically identify subsumption relations, they can also falsely intro-

[2] https://stanfordnlp.github.io/CoreNLP/api.html

Table 5.3: Lexico-syntactic patterns.

Pattern	Example
Including...	type of book *including* monograph, collection, proceeding...
Includes...	a part of an aircraft, this *includes* the engine, wheels, etc.
Such as...	organization *such as* sports organization, governmental organization, etc.
E.g....	different events, *e.g.* sports events, academic events, etc.
For example...	a report, *for example* a research report, technical report, etc.

duce meronymic relations. The challenge of distinguishing subsumption from meronymy is also mentioned in Arnold and Rahm [7]. For example, the natural language definition for the concept *Airport* could include the phrase "*Airports often have facilities to store and maintain aircraft, and a control tower.* ***An airport often includes adjacent utility buildings such as control towers, hangars and terminals***". This would from the patterns above lead to the following subsumption relation: $Airport > Terminal$. However, the correct semantic interpretation of these two concepts would be that a terminal is a part of an airport (not a type/kind of airport), hence these two concepts belong to a meronymic relation (see Section 2.4.3). Therefore, this matcher includes a post-matching operation that checks if the subsumed concept in the candidate subsumption relation belongs to the set of meronyms associated with the subsuming concept. If so, the candidate subsumption relation is discarded. The sets of meronyms are retrieved from WordNet.

5.4 Matcher Selection and Configuration

This section describes how matchers are automatically selected and configured based on the ontology profiling process described in chapter 5.1.

We start with a recap of the ontology profiling described in the previous section. Table 5.4 illustrates how the different ontology profiling metrics determine the selection and configuration of the different matchers. If a matcher is listed in the Matcher Selection column of the table, the measurement of the associated ontology profiling metric determines if the matcher is included in the matcher ensemble or not. If a matcher is listed in the Matcher Configuration column of the table, the measurement of the associated ontology profiling metric is used as a weight parameter to the confidence value each matcher computes for any semantic relation it identifies.

An example is used to explain Table 5.4: The Definition Coverage is calculated by measuring the fraction of concepts in the two input ontologies that have a natural language definition associated to them. The Definition Equivalence Matcher and the Definition Subsumption Matcher both exploit such natural language definitions to infer either equivalence or subsumption relations between ontology concepts. If the ontology profiling determines that the score for the Definition Coverage metric is 0.0, meaning that there are no natural language definitions that can be exploited by the Definition Equivalence Matcher or the Definition Subsumption Matcher, these two matchers should be omitted from the ensemble of matchers run for this particular task. Hence, the ontology profiling metric Definition Coverage is a determinant for the matcher selection process.

Table 5.4: How ontology profiling metrics determine selection and configuration of matcher ensemble

Ontology Profiling Metric	Matcher Selection	Matcher Configuration
Compound Fraction	Compound Matcher	Compound Matcher
Definition Coverage	Definition Equivalence Matcher Definition Subsumption Matcher	Definition Subsumption Matcher
Corpus Coverage	Word Embedding Matcher	Word Embedding Matcher Definition Equivalence Matcher
Property Fraction	Property Equivalence Matcher	Property Equivalence Matcher
Structural Profile	Graph Equivalence Matcher Context Subsumption Matcher	Graph Equivalence Matcher Context Subsumption Matcher
Lexical Coverage	Lexical Equivalence Matcher Lexical Subsumption Matcher	Lexical Equivalence Matcher Lexical Subsumption Matcher

Furthermore, the Definition Subsumption Matcher uses the natural language definitions directly (i.e. without any external support) to arrive to a conclusions on whether two concepts belong to a subsumption relation. Hence, if the Definition Coverage score is 0 or very low, this matcher should be penalised by reducing the confidence measure associated with the relations in the alignment it produces. Therefore we also use the Definition Coverage score as a means for configuring the matcher (i.e. as a weight parameter associated with the confidence of the matcher).

The Definition Equivalence Matcher, however, is based on two sources of input. One is the natural language definition associated with ontology concepts, and the other is the word embeddings generated from the Word2Vec process (see explanation in Section 5.2.1). For each word in the definition of a concept, the Definition Equivalence Matcher does a look-up in the list of word embeddings, and computes a Global Vector that averages the vectors for each individual word in addition to the concept name (label). This means that even without any natural language definitions associated with the ontology concepts to be matched, the Definition Equivalence Matcher can still produce an alignment based on the vectors of the concept names (i.e. a Label Vector as described in Section 5.2.1). From this, it is better to use the Corpus Coverage metric to determine the weight for the confidence value associated with the Definition Equivalence Matcher, and not the Definition Coverage ontology profiling metric.

5.4.1 Strategy for Selecting Matchers based on Ontology Profiling

A matching system's ability to select the most relevant matchers from a possibly large ensemble of matchers is crucial for leveraging a good quality alignment. Recall from Section 5.1 that the result from the ontology profiling process is a set of scores that define the terminological, structural and lexical characteristics of the input ontologies. Similarly, the matchers (both equivalence and subsumption) are also classified as terminological matchers, structural matchers and lexical matchers.

The ontology profiling returns a score normalised between 0 and 1. This score is compared against a predefined *matcher selection threshold* that determines whether a matcher should be included in the matching process or not. In the evaluation described in Chapter 6 we use 0.5 as such a threshold. This means that if a matcher is assigned a score of 0.5 or above, it is included in the ensemble, otherwise it is omitted from the ensemble.

5.4.2 Strategy for Configuring Matchers based on Ontology Profiling

Most approaches for configuring matchers rely on manual intervention or some supervised machine learning method [24, 120, 41]. Configuring or tuning matchers normally requires both domain expertise and in many cases in-depth familiarity with the matching system, hence manual configuration is not always feasible. With regards to supervised learning approaches, these use previously solved matching tasks as training to find effective choices for matcher selecting and configuration [26]. Thus, such an approach require an substantial amount of training data for each individual matching task, something that might be difficult to acquire [120]. Therefore, in this work, the matcher selection and configuration is based on data that can be obtained by profiling the ontologies to be matched.

Initially, the weighting formula in this work is based on the assumption that the higher the profile score, the better are the operating conditions for a matcher that exploits the characteristics for which the profile aims to measure. Based on this assumption, we want to put more trust in matchers that can benefit from “good operating conditions”, and lower the trust in matchers that have to cope with “bad operating conditions”. A straightforward approach would be to simply multiply the confidence value with the profiling score in order to get a weighted confidence value. However, that would result in a too significant reduction of a confidence value assigned by a matcher given a relatively high profile score. In some way we should be able to preserve the best relations (i.e. those with the highest confidence value) even if a matcher is demoted based on the profile score.

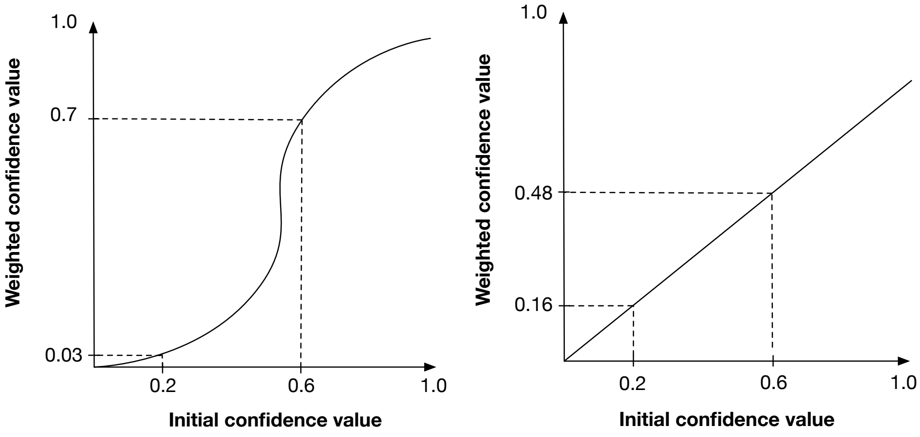
Let us illustrate with an example: a confidence value of 0.8 is intuitively considered a relatively high value. But if the matcher that produced this relation was given a weight of 0.73 from the score obtained in the ontology profiling (which also, intuitively, seems fairly high), this would reduce the confidence value from 0.8 to 0.58. Using the default cut threshold in e.g. the AgreementMakerLight system [39], which is 0.6, this would disregard this relation from the output alignment if a fixed cut-off value was to be used.

Instead, we want a function that leads to a more relaxed demotion of high-confidence relations such that these can be preserved during the alignment combination – even if the matcher is assigned a reduced weight as a result from the ontology profiling. A possible solution to this is the mathematical function *sigmoid*. The sigmoid function creates a curve shaped as an s, in contrast to a linear line, as illustrated in Figure 5.12. Here, we see in Figure 5.12(a) that an initial confidence value of 0.2 (x-axis) is transformed into a weighted confidence value of 0.03 (y-axis) using the sigmoid function, whereas an initial confidence value of 0.6 gets an increase to 0.7. With a linear function as illustrated in Figure 5.12(b) the initial confidence value of 0.2 is reduced to 0.16 and also the initial confidence value of 0.6 is reduced (to 0.48).

The general sigmoid function is defined as described in equation 5.3.

$$\sigma(x) = \frac{L}{1 + e^{-a(x-x_0)}} \tag{5.3}$$

where e is the natural logarithm base, x_0 is the sigmoid’s midpoint along the x-axis, L is the curve’s maximum, and a is called the slope parameter which defines the steepness of the curve.



(a) Weighting using a sigmoid function. **(b)** Weighting using a linear function.

Figure 5.12: Difference between a sigmoid function and a linear function.

Depending on how we want to shape the s curve, the function can demote confidence values considered low and promote confidence values considered high. In other words, using this function enables us to "manipulate" the weight configuration used for transforming an initial confidence value associated with a semantic relation to a weighted confidence value. A similar idea was applied by Ehrig and Sure [28] when they used a fixed midpoint (x_0) of 0.5 to reinforce confidence values above 0.5 and weaken those confidence values below 0.5.

As an attempt to preserve the semantic relations produced under "good operating conditions" quantified using the profile score, as described further above, we modify the original sigmoid function as follows:

$$\sigma(x) = \frac{1}{1 + e^{-a(x - f(PS)))}} \tag{5.4}$$

where in addition to the aforementioned parameters f is a transformation function that transforms the initial profile score (PS) in the range [0.0, ... , 1.0] into some refined range [x, ... , y]. Hence, different from Ehrig and Sure, we have a dynamic midpoint value (x_0) determined by the profile score. This transformation function proceeds as follows:

$$f(x) = (x - b)\frac{e - d}{c - b} + d \tag{5.5}$$

where b is the minimal confidence value, c is the maximal confidence value, d is the minimal value in the output range and e is the maximal value in the output range.

In the following we describe the effect of manipulating the sigmoid parameters. First we focus on the slope parameter a. The choice of slope parameter has an effect on how we discriminate between low vs. high confidence values when combined with the profile score. This is illustrated in Figure 5.13, where we see how initial confidence values are transformed into revised confidence values according to different profile scores. If the slope parameter is set closer to 1 (as indicated by the straight lines in the chart), this has the effect that the transformed confidence from 0.1-1.0 centres in the middle of the y-axis (around 0.5), regardless of which ontology profile score a matcher is assigned. However, if the slope parameter is 20 (as indicated by the dotted lines in the chart), all confidence values below 0.5 are moved very close to zero confidence, while all confidence values above 0.5 are moved very close to confidence 1.0.

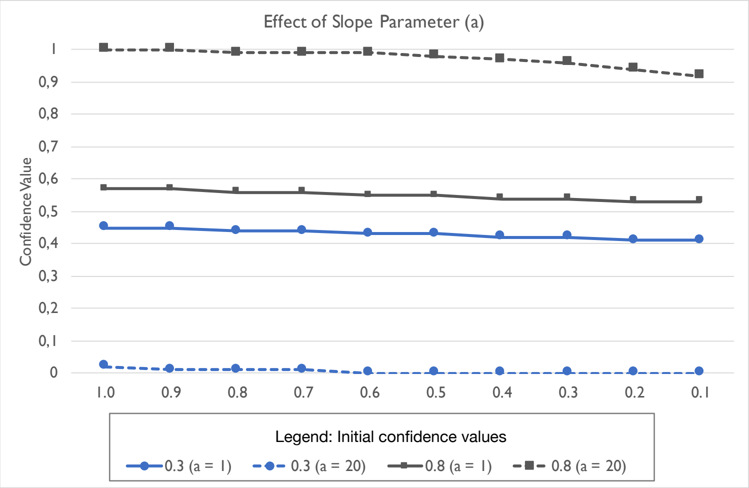


Figure 5.13: Effect of the sigmoid's slope parameter.

Next, we focus the attention on the midpoint parameter x_0 for which we use a transformation function $f(PS)$ to factor in the profile score. The choice of range used for transforming the initial profile score PS is important in order to differentiate between revised confidence scores. This is illustrated in Figure 5.14 where the effect of choosing a range of 0.5 - 0.6 versus 0.5 - 0.7 is demonstrated. Note how an initial confidence value of 0.5 declines more along the x-axis if the initial profile scores are transformed to the range 0.5 - 0.7 (indicated by dotted line in the chart) than if they are transformed to the range 0.5 - 0.6 (indicated by straight line). In this example the slope parameter a is held constant at 12.

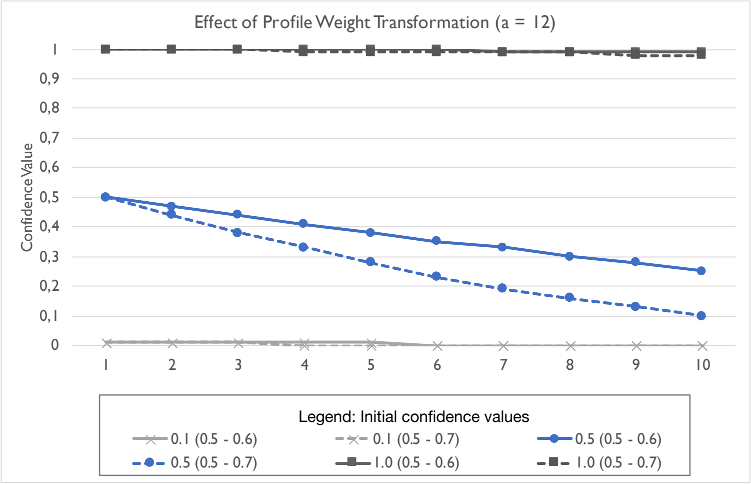


Figure 5.14: Effect of transforming profile weights.

Now that we have explained the effect of manipulating the slope parameter and the midpoint value based on the initial profile score we explain how this will help preserve those relations that fulfil “good operating conditions”. As an attempt to operationalise such conditions three fixed thresholds are applied as input requirements for determining which slope parameter a and which transformation range we should use in the transformation function $f(PS)$. These thresholds are explained in the following.

- *Initial Confidence Threshold*: The initial confidence value associated with the relation should be above some confidence threshold. There is no absolute requirement for what such a threshold should be, as this depends on a number of different aspects (e.g. how the similarity/relatedness scores are computed by a basic matcher). Here, we decided to use the same value as the default confidence threshold in the AgreementMakerLight system, namely 0.6, as our threshold.

- *Profile Score Threshold*: The profile score associated with the matcher should be above a minimum threshold. As mentioned in Section 5.4.1, the matcher selection threshold is set to 0.5, meaning that matchers given a profile score of 0.5 are included in the matcher ensemble. It thus makes sense to use the same threshold as the profile score threshold.

- *Weighted Confidence Threshold*: Finally, the revised confidence value computed using the Sigmoid function should be above threshold. Here, we set 0.5 as our threshold.

An experiment was run to investigate which slope parameter a and transformation range combination would fulfil the requirements posed by the abovementioned thresholds. In this experiment a series of different slope parameters and parameters for the transformation function were used as input.

The results from this experiment are illustrated in Figure 5.15 and Figure 5.16. As Figure 5.15 illustrates, when using a slope parameter of 3 and a transformation range of 0.5-0.7, a relation having an initial confidence value of >= 0.6 would be preserved if it has a profile score of >= 0.5 and a weighted confidence score of >= 0.5. This combination of slope parameter and transformation range fulfils all required thresholds and is therefore used in the prototype semantic matching system in this thesis.

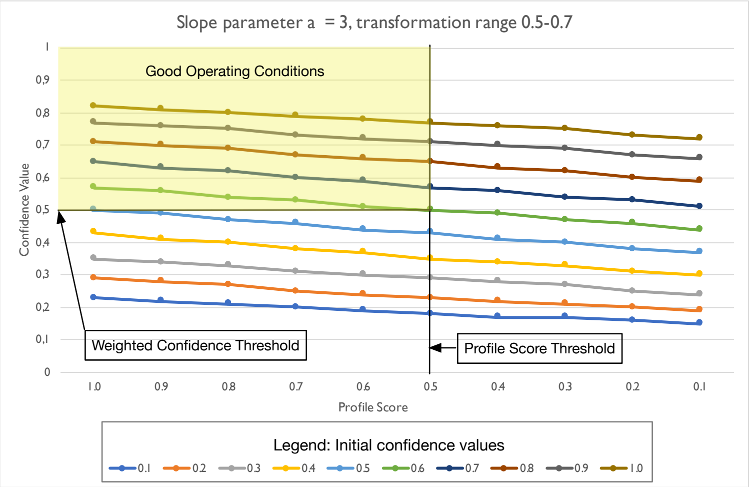


Figure 5.15: Correlation between a confidence value and profile weights *with* the sigmoid function.

Figure 5.16 illustrates the effect of *not* using the sigmoid function and the profile score transformation. Here, the profile score associated with the matcher in question is simply multiplied with the initial confidence score. In such a scenario, only the relations with an initial confidence value of 1.0 would be included using the same thresholds as described above.

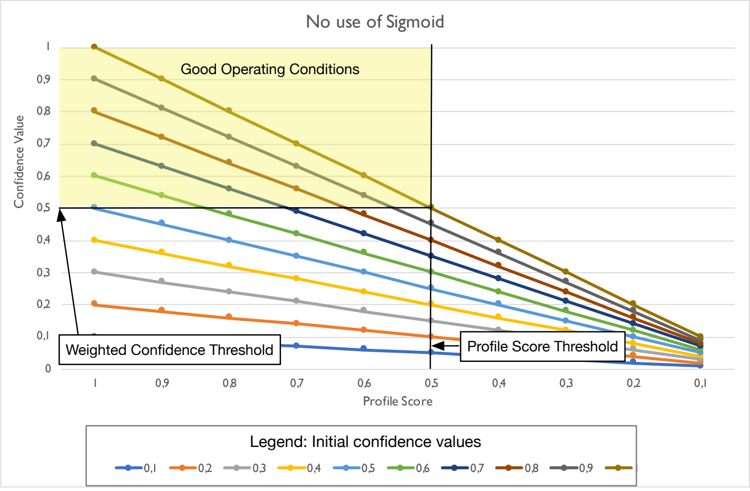


Figure 5.16: Correlation between a confidence value and profile weights *without* the sigmoid function.

5.5 Mismatch Detection Strategies

This section describes two strategies, that inspired by mismatch classification theories (see Section 2.3), aim to automatically detect mismatches that have not been identified as such by matching techniques and therefore are considered true positives. The primary objective of these strategies is to improve the precision of computed equivalence alignments without suffering recall. In the following sub-sections the strategies are explained, and their use is described by two experiments. In these experiments a state-of-the-art matching system, AgreementMakerLight [39] produces an initial alignment from which false positive relations are filtered out using the two strategies.

5.5.1 Concept Scope Mismatch Detection

The Concept Scope Mismatch detection strategy (see Algorithm 3) filters out concept scope mismatches by trying to determine if the matched classes are unlikely to be equivalent because they are in a subsumption or part-whole relationship to one another:

1. Part-whole pattern. The part component of the part-whole relationship includes the name of its whole as its compound modifier. For example, an $AircraftEngine$ represents a part of $Aircraft$. This is classified as a 'component/integral object' relation in Winston et al. [157].

2. Subsumption pattern. The relation is considered a subsumption relation if the compound head in one concept equals the full name of the other (e.g. $Location$ - $ReferenceLocation$). This strategy is also described in Arnold and Rahm [7].

Algorithm 3 starts by iterating over all relations a in the already computed alignment A_{input} (line 3).

Algorithm 3 Technique for filtering out Concept Scope Mismatches

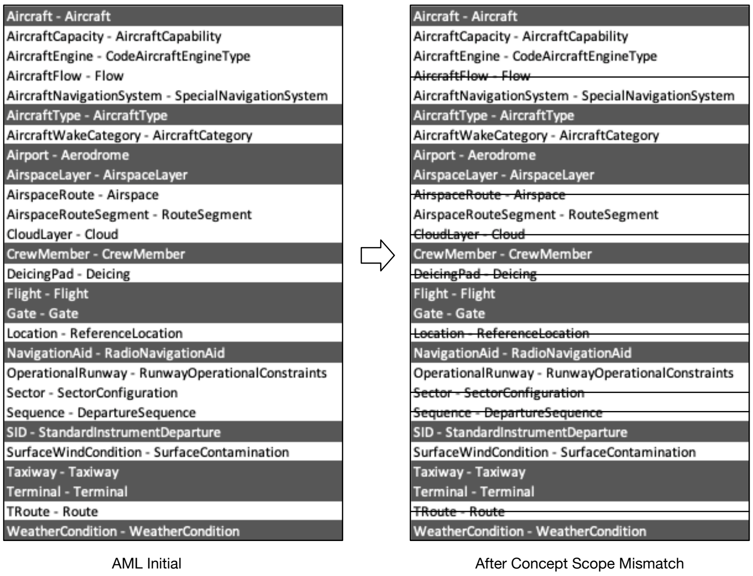
Input: Input alignment A_{input} produced by an ontology matching system.
Output: Alignment A_{output} where relations considered Concept Scope Mismatches are removed.

```
 1: function removeConceptScopeMismatches($A_{input}$)
 2: $A_{filtered} \leftarrow \emptyset$
 3: $A_{output} \leftarrow \emptyset$
 4: for all $a \in A_{input}$ do
 5:     if isCompound($a_{ci}$) then
 6:         $modifier_{a_{ci}} \leftarrow getCompoundModifier(a_{ci})$
 7:         $compoundHead_{a_{ci}} \leftarrow getCompoundHead(a_{ci})$
 8:         if $modifier_{a_{ci}}.equals(a_{cj})$ or $compoundHead_{a_{ci}}.equals(a_{cj})$ then
 9:             $A_{filtered} \leftarrow A_{filtered} \cup a$
10:         end if
11:     end if
12:     if isCompound($a_{cj}$) then
13:         $modifier_{a_{cj}} \leftarrow getCompoundModifier(a_{cj})$
14:         $compoundHead_{a_{cj}} \leftarrow getCompoundHead(a_{cj})$
15:         if $modifier_{a_{cj}}.equals(a_{ci})$ or $compoundHead_{a_{cj}}.equals(a_{ci})$ then
16:             $A_{filtered} \leftarrow A_{filtered} \cup a$
17:         end if
18:     end if
19: end for
20: return $A_{output} \leftarrow A_{input}$ - $\{A_{filtered}\}$
21: end function
```

The $isCompound$ method on line 4 checks if the label of the first class in the relation (a_{ci}) represents a compound. If so, both the compound modifier and the compound head is added to variables $modifier$ and $compoundHead$ respectively (lines 5 and 6). Then the method checks if the label of the other class in the relation (a_{cj}) equals the compound modifier or the extracted compound head. If so, this relation is considered a part-whole relation or a subsumption relation, not an equivalence relation, and is added it to the alignment $A_{filtered}$ which holds all detected concept scope mismatches (line 8). Lines 11-17 follow the same pattern but in the opposite direction to see if the second class in the relation (a_{cj}) represents a part to the whole of the first class in the relation (a_{ci}). Finally, once all relations a are iterated, the detected mismatches are extracted from the input alignment and a refined alignment is returned on line 19.

In order to illustrate the capability of the Concept Scope Mismatch Detection, an experiment was conducted using the AgreementMakerLight [39] ontology matching system. The ontologies from which an alignment has been produced are ATMONTO and AIRM-O, the same two ontologies that are used in the ATM dataset, one of the datasets used for the evaluation in this thesis (see Section 6.2).

Figure 5.17 illustrates how the Concept Scope Mismatch Detection strategy filters out false positive relations from an alignment returned by the AgreementMakerLight system. The relations with dark grey background indicate true positive relations, and the relations that have strike-through text are relations filtered out by the mismatch detection strategy. The result of the mismatch detection in this preliminary experiment was that the precision increased from 0.44 to 0.63 and without any reduction in the recall, the F-measure increased from 0.4 to 0.44.



Aircraft - Aircraft	Aircraft - Aircraft
AircraftCapacity - AircraftCapability	AircraftCapacity - AircraftCapability
AircraftEngine - CodeAircraftEngineType	AircraftEngine - CodeAircraftEngineType
AircraftFlow - Flow	~~AircraftFlow - Flow~~
AircraftNavigationSystem - SpecialNavigationSystem	AircraftNavigationSystem - SpecialNavigationSystem
AircraftType - AircraftType	AircraftType - AircraftType
AircraftWakeCategory - AircraftCategory	AircraftWakeCategory - AircraftCategory
Airport - Aerodrome	Airport - Aerodrome
AirspaceLayer - AirspaceLayer	AirspaceLayer - AirspaceLayer
AirspaceRoute - Airspace	~~AirspaceRoute - Airspace~~
AirspaceRouteSegment - RouteSegment	AirspaceRouteSegment - RouteSegment
CloudLayer - Cloud	~~CloudLayer - Cloud~~
CrewMember - CrewMember	CrewMember - CrewMember
DeicingPad - Deicing	~~DeicingPad - Deicing~~
Flight - Flight	Flight - Flight
Gate - Gate	Gate - Gate
Location - ReferenceLocation	~~Location - ReferenceLocation~~
NavigationAid - RadioNavigationAid	NavigationAid - RadioNavigationAid
OperationalRunway - RunwayOperationalConstraints	OperationalRunway - RunwayOperationalConstraints
Sector - SectorConfiguration	~~Sector - SectorConfiguration~~
Sequence - DepartureSequence	~~Sequence - DepartureSequence~~
SID - StandardInstrumentDeparture	SID - StandardInstrumentDeparture
SurfaceWindCondition - SurfaceContamination	SurfaceWindCondition - SurfaceContamination
Taxiway - Taxiway	Taxiway - Taxiway
Terminal - Terminal	Terminal - Terminal
TRoute - Route	~~TRoute - Route~~
WeatherCondition - WeatherCondition	WeatherCondition - WeatherCondition

AML Initial

After Concept Scope Mismatch

Figure 5.17: Illustration of Concept Scope Mismatch Detection.

5.5.2 Domain Mismatch Detection

Ontology matching systems often rely on some string-based similarity technique for computing similarity between ontology concepts [15]. Unfortunately, the emphasis on string representation of concept names, without any additional semantic analysis, often bring false positive relations in their computed alignment [114], resulting in sub-optimal alignments that possibly propagate errors to the remaining ontology matching workflow. This is a problem that typically increases for ontologies that only partially describe the same domain. Hence, in order to optimise the quality in ontology matching it is therefore important to reduce the number of false positives from string matching operations.

The approach suggested here is based on an assumption that if two concepts are semantically similar, the domains they are associated with ought to be similar too. Following the example illustrated in Figure 5.18, let us say that a string matcher computes with 96 percent confidence that the entity 'Content' in ontology 1 is equivalent with 'Continent' in ontology 2, which is obviously incorrect to the human eye. Such a false positive relation can be filtered out if the similarity between their domains ({Metrology and Photography} vs. {Geography}) is below a given threshold.

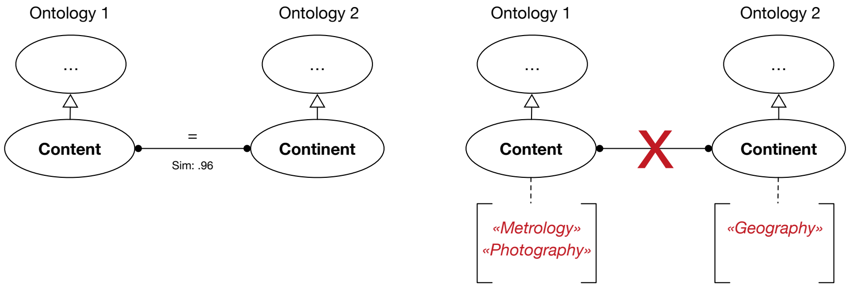


Figure 5.18: Example of applying WordNet Domains to infer domain (dis)similarity.

In this approach WordNet Domains [45], a lexical resource that offers a domain classification of Wordnet synsets (i.e. sets of synonyms for every distinct concept), is applied to determine domain dissimilarity between two concepts. This domain classification is used to verify if the two concepts, or the pre-processed version of them, represent the same domain. If not, the relation holding these two concepts is filtered out of the alignment. The domain dissimilarity is computed using Jaccard similarity of sets on the domains returned as there can be several domains listed for each concept and some domains are used quite generically (e.g. factotum). Of course, the threshold value used for the Jaccard similarity influences on the final result. Experiments using an independent dataset showed that using a similarity threshold of 0.6 and above gave the best results.

The proposed approach to identify false positive relations is illustrated in Algorithm 4. Each relation in this original alignment is processed in a sequence of operations.

Algorithm 4 Technique for detecting Domain Dissimilarity Mismatches

```
Input: Input alignment $A_{input}$ holding a set of relations $a_i$, $minJaccard$ a threshold for Jaccard
    set similarity in the range $[0, 1]$
Output: An alignment $A_{corrected}$ holding relations where the source concept and target concept
    are associated with the same domains.
 1: function $removeDomainMismatches(A_{input})$
 2: $A_{corrected} \leftarrow \emptyset$
 3: for all $a_i \in A_{input}$ do
 4:     if $a_i.c_1$.equals($a_i.c_2$) then
 5:         $A_{corrected} \leftarrow A_{corrected} \cup ai$
 6:     else if $compareConceptNamesDomains(a_i.c_1, a_i.c_2, minJaccard)$ then
 7:         $A_{corrected} \leftarrow A_{corrected} \cup ai$
 8:     else if $fullWordRep(fullWord(a_i.c_1), fullWord(a_i.c_2), minJaccard)$ then
 9:         $A_{corrected} \leftarrow A_{corrected} \cup ai$
10:     else if $compoundHead(compoundWord(a_i.c_1), compoundWord(a_i.c_2), minJaccard)$ then
11:         $A_{corrected} \leftarrow A_{corrected} \cup ai$
12:     else if $compareAllParts(compareAllParts(a_i.c_1), compareAllParts(a_i.c_2), minJaccard)$ then
13:         $A_{corrected} \leftarrow A_{corrected} \cup ai$
14:     end if
15: end for
16: return $A_{corrected}$
17: end function
```

The first operation (line 4) compares the two concept names for string-based equality and there is no interaction with WordNet Domains in this operation. If the concepts are equal, they are added to the revised alignment $A_{corrected}$ without further processing. If they are not, the second operation is initiated.

In operation 2 (`compareConceptNamesDomains`) on line 6, the domains associated with the concept names as they are represented in the ontology are identified, without any text processing involved. So for instance, if one relation includes the source concept “Classification” and the target concept “Class”, the sets of domains associated with these two classes are retrieved. Next, these two sets of domains are compared using Jaccard similarity of sets. If the Jaccard score is equal to or above the $minJaccard$ parameter, the function considers that the two concepts represent the same domain.

Often a concept name is represented as a compound, for instance “TableOfContents”. In operation 3 (`fullWordRep`) on line 8, a compound is split before the interaction with WordNet Domains is performed. So instead of retrieving domains for “TableOfContents” from WordNet Domains, any domains associated with “Table Of Contents” are retrieved. The remaining part of this step is similar to operation 2.

In operation 4 (`compoundHead`) on line 10, the compounds are also split. However, in this step only the compound head (the part that carries the basic meaning of the whole compound) of the concept names is considered. Hence, only domains associated with the compound heads are retrieved, and if the Jaccard score is equal to or above the *minJaccard* threshold, the relation is added to the corrected alignment.

Finally, operation 5 (`compareAllParts`) on line 12 retrieves the domains of all “atomic” words from a compound. So, if for example, a relation includes the source concept “MusicNotation” and the target concept “MusicComposition”, the domains for “Music” and “Notation” are retrieved for the source concept, and the domains for “Music” and “Composition” are retrieved for the target concept. Then the sets for each concept are merged and compared using Jaccard as in the previous operations.

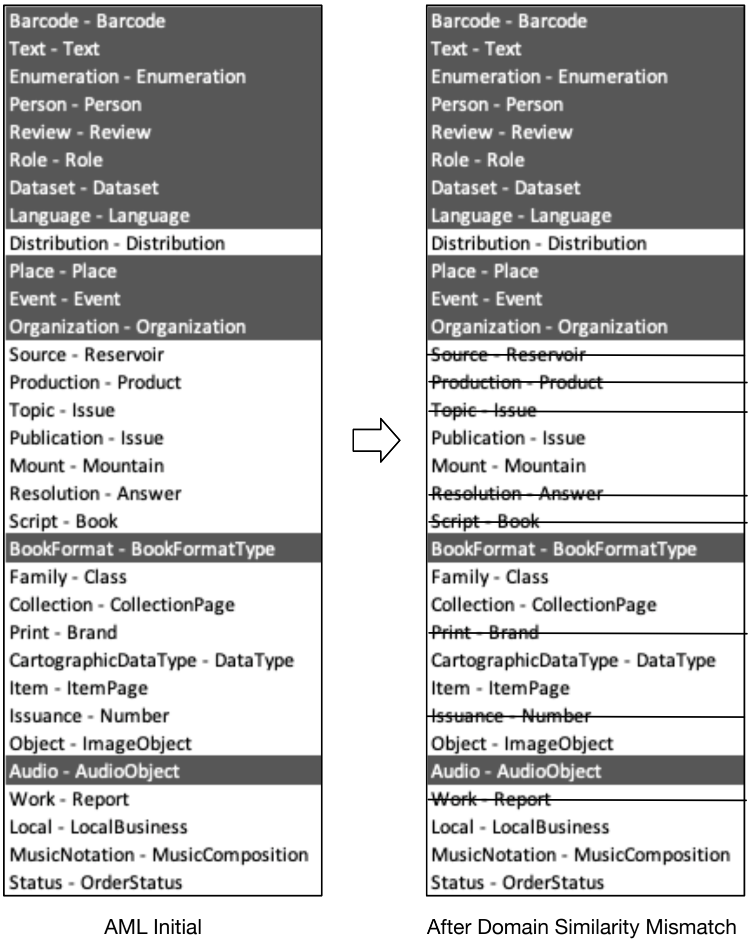
As with the Concept Scope Mismatch Detection presented in the previous sub-section, an experiment using the AgreementMakerLight [39] ontology matching system was conducted. This time the ontologies being matched were from the second dataset used for the evaluation in this thesis, Bibframe and Schema.org. These two ontologies form the Cross-Domain dataset which is presented in detail in Section 6.3.

Figure 5.19 illustrates the effect of this mismatch detection strategy. The relations with dark grey background indicate true positive relations, and the relations that have a strike-through text are relations filtered out by the Domain Mismatch Detection strategy.

In this experiment the precision increased from 0.4 to 0.54, and without any expense of recall, the F-measure increased from 0.54 to 0.65.

5.6 Combining Matcher Results

Four different alignment combination methods have been implemented in order to aggregate relations from alignments produced by the individual matchers presented in previous chapters. Three of them represent a non-weighted approach, i.e. they do not consider any weighted confidence values for the aggregation of relations from the individual alignments. The fourth uses a weighted aggregation approach in that the profile scores computed



Barcode - Barcode
Text - Text
Enumeration - Enumeration
Person - Person
Review - Review
Role - Role
Dataset - Dataset
Language - Language
Distribution - Distribution
Place - Place
Event - Event
Organization - Organization
Source - Reservoir
Production - Product
Topic - Issue
Publication - Issue
Mount - Mountain
Resolution - Answer
Script - Book
BookFormat - BookFormatType
Family - Class
Collection - CollectionPage
Print - Brand
CartographicDataType - DataType
Item - ItemPage
Issuance - Number
Object - ImageObject
Audio - AudioObject
Work - Report
Local - LocalBusiness
MusicNotation - MusicComposition
Status - OrderStatus

AML Initial

Barcode - Barcode
Text - Text
Enumeration - Enumeration
Person - Person
Review - Review
Role - Role
Dataset - Dataset
Language - Language
Distribution - Distribution
Place - Place
Event - Event
Organization - Organization
~~Source - Reservoir~~
~~Production - Product~~
~~Topic - Issue~~
Publication - Issue
Mount - Mountain
~~Resolution - Answer~~
~~Script - Book~~
BookFormat - BookFormatType
Family - Class
Collection - CollectionPage
~~Print - Brand~~
CartographicDataType - DataType
Item - ItemPage
~~Issuance - Number~~
Object - ImageObject
Audio - AudioObject
~~Work - Report~~
Local - LocalBusiness
MusicNotation - MusicComposition
Status - OrderStatus

After Domain Similarity Mismatch

Figure 5.19: Illustration of Domain Mismatch Detection.

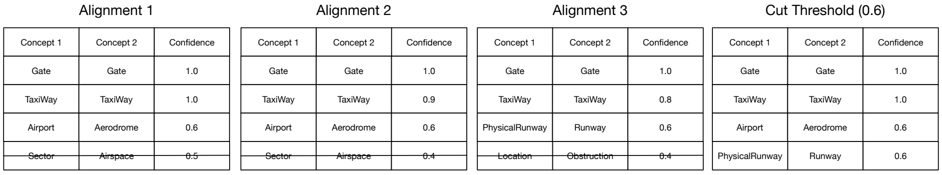
in the Ontology Profiling process are used to weight the confidence values assigned by the matchers.

The combination methods based on a non-weighted aggregation approach are *Cut Threshold*, *Average Aggregation*, and *Majority Vote*. The weighted aggregation approach is named *Profile Weight*. In this chapter all methods are described in detail, and all of them will later be evaluated in Chapter 6.

5.6.1 Cut Threshold

The Cut Threshold combination method is based on only allowing the relations from the individual alignments having a confidence value at or above a given threshold access to the final alignment. As the example in Figure 5.20

shows, only the relations from the individual alignments having a confidence value over 0.6 are included in the final alignment to the right.



Alignment 1

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	1.0
Airport	Aerodrome	0.6
~~Sector~~	~~Airspace~~	~~0.5~~

Alignment 2

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	0.9
Airport	Aerodrome	0.6
~~Sector~~	~~Airspace~~	~~0.4~~

Alignment 3

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	0.8
PhysicalRunway	Runway	0.6
~~Location~~	~~Obstruction~~	~~0.4~~

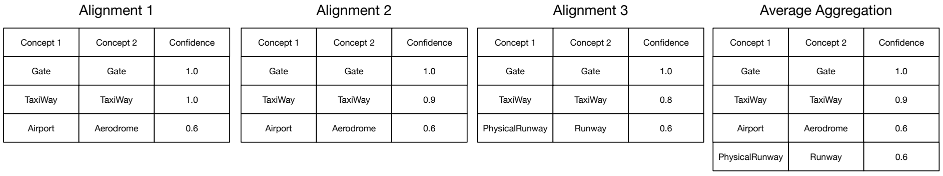
Cut Threshold (0.6)

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	1.0
Airport	Aerodrome	0.6
PhysicalRunway	Runway	0.6

Figure 5.20: Cut Threshold of Matcher Alignments.

5.6.2 Average Aggregation

The Average Aggregation combination method averages the confidence of relations where the two concepts involved are the same across all alignments. As the example in Figure 5.21 shows, this method adds all relations that are included in the individual alignments, and creates a final confidence score based on the average confidence of the individual alignments. This computation is done regardless of the type of relation that holds between the two concepts in a relation.



Alignment 1

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	1.0
Airport	Aerodrome	0.6

Alignment 2

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	0.9
Airport	Aerodrome	0.6

Alignment 3

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	0.8
PhysicalRunway	Runway	0.6

Average Aggregation

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	0.9
Airport	Aerodrome	0.6
PhysicalRunway	Runway	0.6

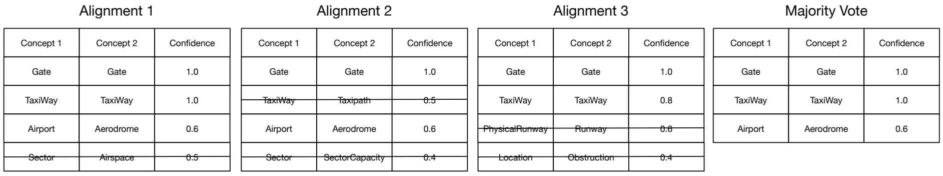
Figure 5.21: Average Aggregation of Matcher Alignments.

5.6.3 Majority Vote

This combination method implements a voting strategy for determining the final alignment from a set of alignments produced by individual matchers. Here, the relations that are represented in the majority of the individual alignments are included in the final alignment as illustrated in Figure 5.22.

5.6.4 Profile Weight

When aggregating the semantic relations computed by the different matchers into a final alignment the ProfileWeight strategy takes a weighted aggregation approach. This strategy is similar to the alignment aggregation methods by Mao et al. [91] and Gulic et al. [56] described in Section 3.4 in that a part of the strategy involves extracting the "highest relations" from



Alignment 1

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	1.0
Airport	Aerodrome	0.6
~~Sector~~	~~Airspace~~	~~0.5~~

Alignment 2

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
~~TaxiWay~~	~~Taxipath~~	~~0.5~~
Airport	Aerodrome	0.6
~~Sector~~	~~SectorCapacity~~	~~0.4~~

Alignment 3

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	0.8
~~PhysicalRunway~~	~~Runway~~	~~0.6~~
~~Location~~	~~Obstruction~~	~~0.4~~

Majority Vote

Concept 1	Concept 2	Confidence
Gate	Gate	1.0
TaxiWay	TaxiWay	1.0
Airport	Aerodrome	0.6

Figure 5.22: Majority Vote to determine a final alignment from Matcher Alignments.

a similarity matrix. However, the commonly used principles for alignment combination used in equivalence matching are not transferable into a setting where subsumption alignments are to be combined. The reason for this is that these principles operate under the condition that from the set of individual alignments a single best 1-1 relation should be extracted from the individual alignments and be represented in a final alignment.

Contrary to Mao et al. and Gulic et al., the computation of the final alignment is not based upon how many "highest correspondences" there are over all computed relations by each matcher, but rather the profiling weights imposed by the ontology profiling process. Hence, in the Profile Weight combination strategy *the set of* "highest correspondences" are extracted based on the confidence score in each row and each column of a similarity matrix, then from this set of relations the weights from the ontology profiling (according to the description presented in Section 5.4) are imposed before computing the final alignment.

Furthermore, while 1-1 relations are enforced when computing the equivalence alignments, that constraint is removed in the aggregation of subsumption relations since it is highly likely that a single concept in one ontology can be related to multiple concepts in the other ontology. For the equivalence alignments, the Naive Descending Extraction algorithm [94] is applied to enforce 1-1 relations, however this is not performed for the subsumption alignments.

The procedure is described in pseudocode in Algorithm 5. The input to the procedure is a set of alignments R holding relations computed by individual matchers. For each alignment a similarity matrix M is created (Line 4). The set of highest relations (those relations having the highest confidence value row-wise and column-wise) are extracted into set H (Line 5). Finally, the highest relations in H are unionised into a final alignment A_{pw}. If the alignments hold equivalence relations, the method $enforceSM$ ensures that non 1-1 relations are removed using the Naive Descending Extraction algorithm.

Algorithm 5 Pseudocode for the Profile Weight Combination Method

Input: R, a set of alignments produced by individual matchers where the confidence of each relation is weighted based on the scores from the ontology profiling.

Output: A final alignment A_{pw} holding the highest relations from the individual matchers.

```
function $ProfileWeight(O_s, O_t)$
$A_{pw} \leftarrow \emptyset$
for all $A \in R$ do
    M $\leftarrow$ $createSimMatrix(A)$
    H $\leftarrow$ $extractHighestRelations(M)$
end for
return $A_{pw} \leftarrow H$ - $\{enforceSM(H)\}$
end function
```

5.6.5 Merging Equivalence and Subsumption Alignments

As the computation of equivalence relations and subsumption relations is performed separately for the purpose of a more detailed evaluation, the final alignments must be merged in order to finally return a complete alignment holding both equivalence and subsumption relations. This merging process basically takes the union of all relations in the equivalence alignment and subsumption alignment and puts the relations in a merged equivalence and subsumption alignment. However, conflicts can occur if an equivalence relation and a subsumption relation have the same source and target concepts. In such a case the merging process retains the relation with the highest confidence value. If both conflicting relations have the same confidence value, we leave both relations in the final alignment (e.g. for human evaluation).

5.7 Use of External Sources

A number of external sources and libraries are used to support the matching operations. An overview of how the different matchers (described in detail in Sections 5.2 and 5.3) depend on these sources and libraries is presented in Figure 5.23. The OWL API and the Alignment API have been left out of the illustration since they are used by all matchers.

5.7.1 Word Embedding

As described in Section 2.5.1, Word Embedding represents a set of techniques where words (or phrases) are mapped to vectors represented by real numbers. The implementation in this work is based on the Skip-Gram model proposed by Mikolov et al. [96] which aims at predicting surrounding words given a target word. It does so by training a neural network using a large-scale corpus containing sentences of words. Early experiments with

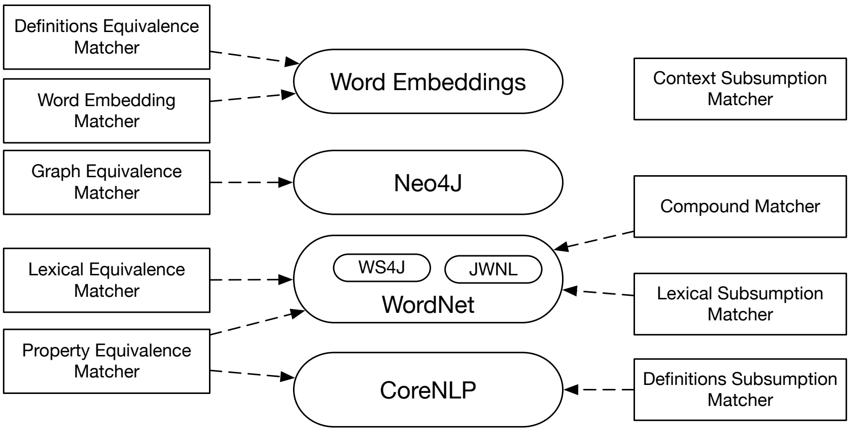

Definitions Equivalence Matcher
Word Embedding Matcher
Graph Equivalence Matcher
Lexical Equivalence Matcher
Property Equivalence Matcher
Word Embeddings
Neo4J
WS4J
JWNL
WordNet
CoreNLP
Context Subsumption Matcher
Compound Matcher
Lexical Subsumption Matcher
Definitions Subsumption Matcher

Figure 5.23: Dependencies to external sources and libraries.

the Skip-Gram and the Continuous Bag of Words (CBOW) architectures revealed little difference in performance, and the rationale for using Skip-Gram over CBOW is mainly that Skip-Gram allegedly performs better on lower-sized corpora. Especially one of the corpora used in this work (the Skybrary corpus described next) has a relatively small initial size (around 40MB).

Two different corpora have been prepared for the Word Embedding Matcher and the Definitions Equivalence Matcher in this work:

- *SKYbrary Corpus.* This corpus is extracted from a wiki called SKYbrary. SKYbrary is an electronic repository of safety knowledge related to flight operations, air traffic management (ATM) and aviation safety in general.
- *Wikipedia Corpus.* This is a more general corpus represented by a complete dump of Wikipedia[3].

For the Wikipedia Corpus we used a dump of the English Wikipedia offered by WikiMedia[4]. For the SKYbrary Corpus the SKYbrary wiki was scraped using the open source tool Dumpgenerator[5]. Both these dumps are represented in large XML files. From thereon, both corpora were transformed to plain text using the WikiExtractor tool[6] and pre-processed by removing all

[3] This corpus was prepared by preprocessing the Wikipedia dump of 18 October 2018.

[4] https://dumps.wikimedia.org/

[5] https://github.com/WikiTeam/wikiteam/blob/master/dumpgenerator.py

[6] https://github.com/attardi/wikiextractor

XML/HTML tags, punctuation, and stopwords; lowercasing all words, and finally tokenizing the text into sentences.

For training the neural network we used a vector dimension of 300, a window-size of 5, and a minimum word count of 2. The Skip-gram algorithm was run for 5 iterations, resulting in a text file holding every word in the initial corpus mapped to a 300-dimensional vector positioning (embedding) this word in a vector space.

5.7.2 WordNet

WordNet [98] is a lexical database where nouns, verbs, adjectives and adverbs are grouped into sets of synonyms (synsets) expressing distinct concepts. The synsets are interlinked by lexical-semantic relations forming a network of related words and concepts. A more comprehensive description of WordNet is provided in Section 2.5.3. WordNet is applied by the Lexical Equivalence Matcher, the Lexical Subsumption Matcher, the Property Equivalence Matcher and the Compound Matcher.

5.7.3 Software Libraries

The software development involved when developing the artefacts described in this chapter has benefited from several existing software APIs and libraries.

The OWL API [66] is a Java reference implementation for creating, manipulating and serialising ontologies in the OWL format. This API has primarily been used for parsing and extracting various features from the ontologies to be matched, both in the ontology profiling process and when performing the actual semantic matching.

The Alignment API [23], which is also described in detail in Section 2.2.2, offers a programming infrastructure for ontology matching and a standardised format for expressing and evaluating alignments. Related to the Alignment API is the OntoSim[7] library which offers various types of similarity techniques especially targeted for semantic matching.

Stanford's Simple CoreNLP[8] is an API for natural language processing. This API is used for Part Of Speech (POS) tagging, sentence tokenisation of ontology concept definitions and lemmatisation of ontology concept names. CoreNLP is used by the Property Equivalence Matcher and the Definitions Subsumption matcher.

[7] http://ontosim.gforge.inria.fr/

[8] https://stanfordnlp.github.io/CoreNLP/simple.html

Two different libraries are used for programmatically accessing the WordNet lexicon. Java WordNet Library (JWNL)[9] offers access to the core services offered by WordNet (e.g. retrieving synonyms and hyponyms) while WordNet Similarity for Java (WS4J)[10] provides a set of similarity techniques that employ the lexical-ontological organisation of concepts offered by WordNet.

The Neo4J [11] graph database is used by the Graph Equivalence Matcher.

[9] https://sourceforge.net/projects/jwordnet/
[10] https://code.google.com/archive/p/ws4j/
[11] https://neo4j.com/

6 Evaluation

The artefacts, that is, profiling metrics, matchers, alignment combination methods, mismatch detection strategies and the resulting prototype, are evaluated using three diverse datasets. These datasets involve pairs of ontologies representing different application domains, size, and complexity. The evaluation is conducted according to the research approach and more general evaluation guidelines specified in Part II.

6.1 Evaluation Protocol

All evaluations are run on a machine with Intel Core i7-7567U processor (3.5 GHz, dual-core) and 16 GB of RAM memory.

As described in the Research Approach in Part II material from the experiments run in the evaluation is made available on-line.

The evaluation metrics precision and recall, semantic precision and recall, and F-measure, as they are described in Section 2.2.3, are used to evaluate artefacts.

Although optimalisation of run-time performance is clearly an important quality of semantic matching systems, this has not been a priority in this work, and is not considered in the evaluation. But in general terms, ontology matching is a problem of quadratic complexity if it involves comparing all concepts in one ontology with all concepts in the other [37], as done in this work. By reducing the search space for the matching process through decomposing the matching task (i.e. the input ontologies) into smaller subtasks (modules) this can reduce the complexity. This is considered out of scope in this work, but for the interested reader we refer to Jiménez-Ruiz et al. [76], who implemented a modularisation approach combining locality modules and neural embeddings, and Hu et al. [69] who implemented an approach that partitions the input ontologies into clusters based on structural characteristics in the Falcon-AO matching system.

6.1.1 Evaluation of Individual Matchers

In each dataset we first evaluate how the individual matching algorithms perform against the ground truth represented by the reference alignments. No matcher weight is applied on these alignments, but 1-1 relations is enforced for equivalence alignments using the Naive Descending Extraction algorithm [94]. The performance of the individual matching algorithms is evaluated using precision, recall and F-measure at different confidence thresholds in order to get a good overview of their performance. Moreover, the complementarity of the different matching algorithms is determined based on an analysis of how many and which of the true positive relations each of them has identified.

6.1.2 Evaluation of Alignment Combination Methods

When evaluating the combination of the alignments produced by the individual matchers, we compare a *weighted approach* against a *non-weighted approach*. In order to properly compare the weighted and non-weighted approach, the exact same process is used for producing a final alignment which is then evaluated. As with the evaluation of the individual matchers, the different alignment combination methods are evaluated on precision, recall and F-measure at different thresholds.

For the weighted approach we evaluate both how the ontology profiling influences the quality of the returned alignment with matcher selection and without matcher selection. The objective here is to see if there is any quality improvement from omitting matchers whose associated profile score (see Section 5.4) is below a *selection threshold*. A selection threshold of 0.5 is used in all datasets.

Weighted Approach

The weighted approach uses the profile scores as follows:

- *Transform initial confidence.* The profile weight relevant for a given equivalence or subsumption matcher is transformed to the range 0.5 - 0.7 according to the function described in Section 5.4.2. A constant slope parameter a of 3 is used together with the function described above to transform the initial confidence value to a final confidence value for each relation computed by each matcher.

- *Combine alignments using Profile Weight.* Once all equivalence and subsumption alignments are computed they are separately combined using the Profile Weight combination method described in Section 5.6.4. This is performed in parallel since the relation aggregation for equivalence and subsumption relations requires different handling as described in Section 5.6.4. The result of this process is one combined alignment file holding equivalence relations and another alignment file holding subsumption relations.
- *Merge equivalence and subsumption alignments.* As a final step the equivalence alignment and the subsumption alignment are merged into a final alignment holding both types of relations.

Non-Weighted Approach

The non-weighted approach to which the weighted approach described above is compared against follows the following steps:

- Non-weighted variants of the equivalence and subsumption alignments produced by the individual matchers are combined using the combination methods *Cut Threshold*, *Average Aggregation* and *Majority Vote*. These methods are described in Section 5.6. As earlier mentioned, except for the weighting, the alignments are produced in exactly the same manner (including the enforcement of 1-1 relations and mismatch detection of the equivalence alignments and the conflict resolution of the subsumption alignments) as for the weighted combination approach described above.
- Once merged according to the approach described in Section 5.6.5, the combined alignments produced by the combination methods described in the previous step are evaluated at all thresholds from 0.1[1] to 1.0 in order to identify the best configuration.

6.1.3 Evaluation of Mismatch Detection Strategies

The mismatch detection strategies (see Section 5.5) are evaluated on the basis of the improvement they achieve on the combined equivalence align-

[1] Evaluating alignments cut at threshold 0.0 does not make sense due to way too many false positives resulting in a very low precision.

ments. Specifically, since the mismatch detection strategies target improvement of precision on equivalence relations (ideally without any compromise in recall), the change of F-measure by running them on the combined equivalence alignment produced by the Profile Weight combination method is measured.

The mismatch detection strategies are run in the following order:

1. Concept Scope Mismatch Detection
2. Domain Mismatch Detection

6.1.4 Comparison with other Matching Systems

To see how the Profile Weight approach performs in the context of other matching systems, this section include two evaluations. The first evaluation compares the performance of systems that output both equivalence and subsumption relations in their alignments. Here, the results from the Profile Weight approach is compared with the results from S-Match [46], STROMA [7], and BLOOMS [71]. For this evaluation traditional precision, recall and F-measure scores are used to indicate the performance of the systems. Since the confidence scoring may vary and the optimal confidence thresholds are not known for all compared systems, the evaluation scores are computed and presented at all confidence thresholds 0.1 - 1.0 to give an as complete view as possible.

The second evaluation includes, in addition to those systems described above, two matching systems that only return alignments holding equivalence relations, namely AgreementMakerLight [39] and LogMap [78]. In this evaluation we also include only the equivalence component of Profile Weight. This will give an indication on how the equivalence matchers described in Section 5.2 and their combination using the Profile Weight combination approach perform relative to state-of-the-art matching systems. Furthermore, this will also indicate how the performance of the subsumption matchers included in Profile Weight compares to simply inferring such relations using a reasoner. For this evaluation semantic precision, recall and F-measure scores which consider inferred relations from the alignments, the reference alignments, and the input ontologies, are used to measure system performance. Semantic precision and recall are computed according to Euzenat's definitions described in Section 2.2.3, while F-measure computes a score that harmonises the two, as with traditional precision and recall. The implementation of Euzenat's semantic precision and recall provided by the Alignment API is used to compute the scores.

There are a few aspects related to some of these systems that should be mentioned:

- S-Match does not come with a parser for OWL ontologies. Therefore, the ontologies used in these three datasets had to be converted into a format accepted by S-Match (basically a text file that maintained the subsumption hierarchy represented in the ontologies) before the matching operations were run.

- As described in Section 3.2 there are three different configurations of S-Match. In this evaluation we have used the *Minimal Semantic Matching* feature. The main reason for choosing this configuration is that it returns direct subsumption relations (from which other transitive subsumption relations later can be derived). This is important in order to have a similar baseline for comparison. Both the subsumption matchers developed in this work as well as reference alignments for all three datasets are based on direct subsumption relations.

- As described in Section 3.2, STROMA is a system that on the basis of an already produced equivalence alignment identifies subsumption, meronymy or "relatedness" relations. A natural equivalence matching system to use in order to produce equivalence alignment is COMA, which is developed by the same research group that has developed STROMA. However, COMA was not able to complete the equivalence matching in the ATM dataset since it ran out of memory when parsing the AIRM-O ontology. For the ATM dataset the AgreementMakerLight [39] system was used to compute the equivalence alignment used as input to STROMA. We used the following configuration settings of AgreementMakerLight: A similarity threshold of 0.5, all available matchers and the Obsolete Filter, the Cardinality Filter, and the Coherence Filter. It should be noted that in the paper describing STROMA [7]), the authors stress that STROMA can work with different matching systems producing the initial equivalence alignment.

- The BLOOMS system does not come with a Graphical User Interface as the other two systems, but its source code is openly available. Some of its source code had to be slightly re-factored in order for the system to work. However, when examining its source code some of its post-processing functionality, *after the alignment files are produced*, relies on web search using a deprecated version of the Bing search framework. As a result of this, there are some reliability concerns with respect to the results from BLOOMS in this evaluation.

- When running LogMap the latest version found on GitHub at `https://github.com/ernestojimenezruiz/logmap-matcher` was used. Furthermore, the configuration file "parameters.txt" was edited to not consider property matching and instance matching in order to be consistent with the other systems.
- AgreementMakerLight was run using the latest version found on GitHub at `https://github.com/AgreementMakerLight` and the default configuration option was used.

6.2 Dataset 1 - Air Traffic Management

This dataset includes two ontologies from the Air Traffic Management (ATM) domain. It partly originates from a research project called BEST (Achieving the BEnefits of SWIM by making smart use of Semantic Technologies)[2] which is also described in the research approach in Part II.

The development of this dataset is described in Gringinger et al. [53] and Vennesland et al. [152].

6.2.1 Dataset Summary

The first ATM ontology in this dataset is the NASA Air Traffic Management Ontology (ATMONTO) [81, 80]. ATMONTO supports semantic integration of ATM data being collected and analysed at NASA for research and development purposes. It includes a wide range of classes and properties covering aspects of flight and navigation, aircraft equipment and systems, airspace infrastructure, meteorology, air traffic management initiatives, and other areas.

The second ontology in this dataset is called AIRM-O, which was developed in the European Project BEST [151]. This OWL ontology was transformed from a semantic reference model for the ATM domain called AIRM (ATM Information Reference Model) according to transformation rules specified by the Object Management Group (OMG) [108]. AIRM provides a common reference for the operational terminology and the data models that have been developed and will continue to be developed to ensure a modern ATM system. It is derived from multiple information exchange models that have been established as standards for the global aviation community:

- The Aeronautical Information Exchange Model (AIXM) [29] stand-

[2] http://www.project-best.eu/

ardises exchange of data pertaining to relatively static aeronautical infrastructure resources, including air routes, airspaces, aerodromes, etc.

- The Flight Information Exchange Model (FIXM) [30] standardises exchange of flight information between ATM systems.
- The ICAO Meteorological Information Exchange Model (IWXXM) [32] is a model that standardises weather information exchange between ATM systems.

The AIRM-O ontology was used in combination with semantic reasoning for supporting retrieval and filtering of ATM information in order to facilitate improved information exchange as envisioned by the System Wide Information Management (SWIM)[3] concept in ATM.

Table 6.1 describes some key metrics related to the ATMONTO and AIRM-O ontologies.

Table 6.1: Ontology statistics for the ATM Dataset

	Classes	Object Properties	Data Properties	Axioms
ATMONTO	157	126	189	2483
AIRM-O	915	1761	494	28408

In order to develop a reference alignment between these two ontologies a panel of five persons, all with experience from the ATM domain and semantic technologies, collaboratively produced a mapping between the two ATM ontologies. Each person was asked to match each of the 157 classes in ATMONTO to corresponding classes in the larger AIRM-O, making use of their domain knowledge as well as all available input including descriptive class and property annotations in the ontologies plus any other informative web resources. In addition to identifying equivalent classes, each person also indicated less/more general relationships between concepts as well as potential mismatches of varying degree. This activity resulted in a detailed classification of mismatch types with the following mismatch types:

- *Differing level of abstraction*: The matched classes intersect, but some instances fall outside the intersection.

[3] https://www.eurocontrol.int/swim

- *Differing scope*: No matching class is present because the class in the source ontology is outside the defined scope of the target ontology.
- *Differing level of detail*: One class is modeled in more depth and with greater fidelity than the other.
- *Differing representation*: The matched classes represent the same concept, but are modeled differently across ontologies and implemented using different representational approaches.
- *Differing intended use*: The matched classes are modeled using significantly different properties and relationships reflecting differences in how the classes are to be used in the context of a domain application.
- *Differing standards*: The matched classes have similar names but define different versions of the concept based on differing technical standards adopted by ontology developers, e.g., by FAA[4] and EUROCONTROL [5].
- *Abstract concept*: The class in the source ontology represents a highly abstract notion that is unlikely to map to a similar class in the target ontology.
- *Differing word senses*: The classes have an exact or close lexical match, but the two classes correspond to two different word senses.
- *Ill-conceived class*: Due to lack of information, time, or resources, not all ontology classes will be properly designed and therefore difficult to match.

These mismatch types emerged independently from the ontology mismatch classifications presented in Section 2.3, but as illustrated in Figure 6.1, which is an adaptation of Klein's [83] original classification presented in section 2.3, there is a close correspondence between most of the mismatch categories.

After the initial matches were compiled, two of the five persons in the panel reviewed the matches for each ATMONTO class and produced a consensus mapping holding equivalence relations between classes from these two ontologies. During this activity, a substantial amount of the previously identified

[4] The Federal Aviation Administration (FAA) regulates all aspects of civil aviation in the U.S.

[5] Eurocontrol is the central organisation for coordination and planning of air traffic control for all of Europe.

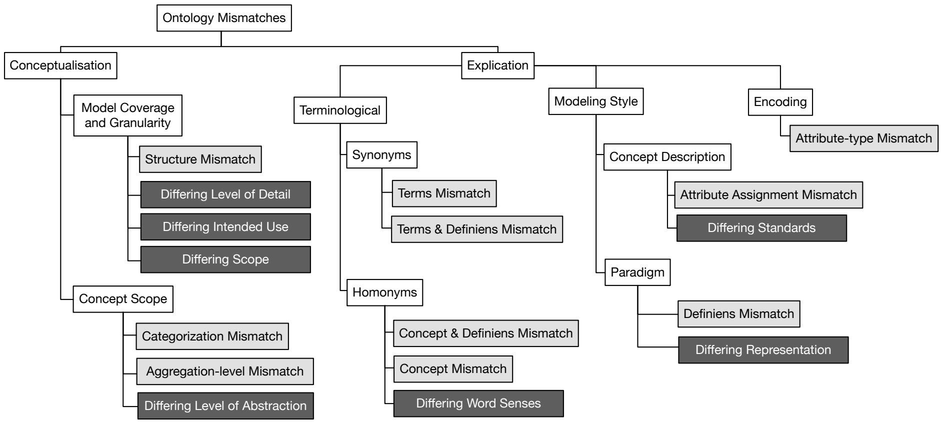


Figure 6.1: Illustration showing how the mismatches derived from the mapping of ATM ontologies relate to ontology mismatch classification from literature.

equivalence relations were considered a “light match” instead of an “exact match”. In these cases, an equivalent relation was deemed too strong, and often the result of the two concept names being equal in effect of their string representation. Identical naming, however, was no guarantee of a correct match between two classes. In fact, in approximately 25 percent of the identified exact match pairs, the two class names did not have any words in common, while in approximately 40 percent of the identified “light match” pairs, the class names did have words in common.

The result of this activity was a set of exact match relations including truly equivalent concepts, and with this as a starting point, the equivalence and subsumption reference alignments were developed using the following approach:

1. Develop equivalence reference alignment based the set of “exact match” relations described above. This was accomplished by transforming this set of relations to an alignment formatted as RDF/XML according to the Alignment Format[6].

2. Infer subsumption relations from equivalence relations in (1) to obtain a subsumption alignment. Here, the same procedure as in OAEI 2011 was followed: The two source ontologies were merged into one single ontology using the ontology editor Protégé[7]. Then *equivalentClass* axioms consistent with the mapping described above were manually

[6] http://alignapi.gforge.inria.fr/format.html
[7] https://protege.stanford.edu/

added between the corresponding classes in the merged ontology. A reasoner (HermiT[8]) was run to classify the classes in the merged ontology in order to infer the subsumption relations. In addition, those less/more general relations that were indicated in the manual mapping process, but not identified by the reasoner, were included in the reference alignment.

3. Manually evaluate the complete reference alignment holding both equivalence relations and subsumption relations

Only direct subsumption relations were considered in the subsumption reference alignment.

6.2.2 Evaluation of Individual Matchers

The individual matchers evaluated in this section are run without any weight from the ontology profiling.

Evaluation of Individual Equivalence Matchers

Figure 6.2 shows the precision, recall and F-measure scores for the individual equivalence matchers. The best performance is achieved by the Lexical Equivalence Matcher (LEM) which at confidence threshold 0.9 obtains an F-measure of 0.44. The alignment produced by LEM includes a total of 13 relations, of which 10 are correct. 1 of the 3 false positives is a subsumption relation ($InternationalAirport < Aerodrome$) while the other two are wrongly included based on similarity in the compound heads (e.g. $AirspaceRouteSegment = RouteSegment$).

The Word Embedding Matcher (WEM) and the Definition Equivalence Matcher (DEM) both obtain an F-measure above 0.3 at confidence threshold 0.5. WEM obtains a high precision, returning an alignment that includes 7 relations, of which 6 are correct. The false positive relation is between the concepts $STAR$ and DME which from the embedding vectors associated with these concepts yield a cosine score of 0.57.

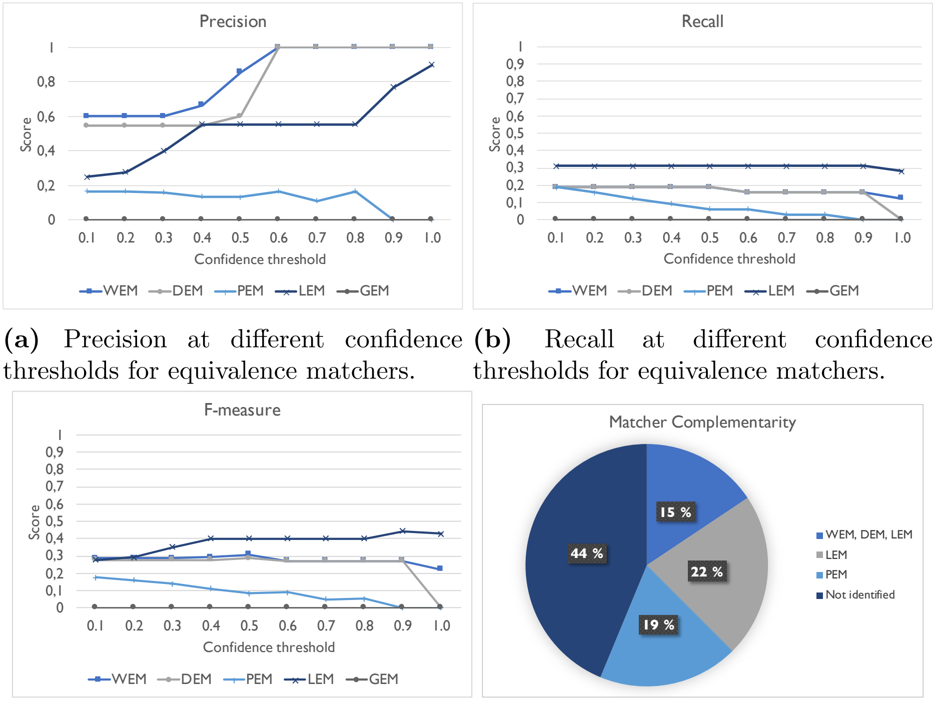
DEM includes the same correct relations as WEM, but gets a lower precision due to additional false positives.

The Property Equivalence Matcher (PEM) obtains its highest performance in the lower thresholds (< 0.5) and then declines. At confidence threshold 0.1 its alignment includes 36 relations, with 6 true positive ones. 5 of these

[8] http://www.hermit-reasoner.com/

are not discovered by any of the other equivalence matchers, hence the contribution from PEM is important once all relations are aggregated into a final alignment. An observation worth noting is that several of the true positive relations from this matcher are given low confidence values. For example, the true positive relation *DeicingPad* = *DeicingArea* is given a confidence value of 0.16. One implication of this is that if alignment combination methods such as Cut Threshold (see Section 5.6.1) are used, this would likely disregard these relations when a final alignment is computed.

The Graph Equivalence Matcher (GEM) does not identify any true positive equivalence relations in this dataset and only includes a false positive in its alignment. This false positive relation (*RadialRoute* = *HoldingArea*) is wrongly inferred from the high string similarity between these two concepts' parents (*AirspaceRoute* and *Airspace*).



(a) Precision at different confidence thresholds for equivalence matchers.

(b) Recall at different confidence thresholds for equivalence matchers.

(c) F-measure at different confidence thresholds for equivalence matchers.

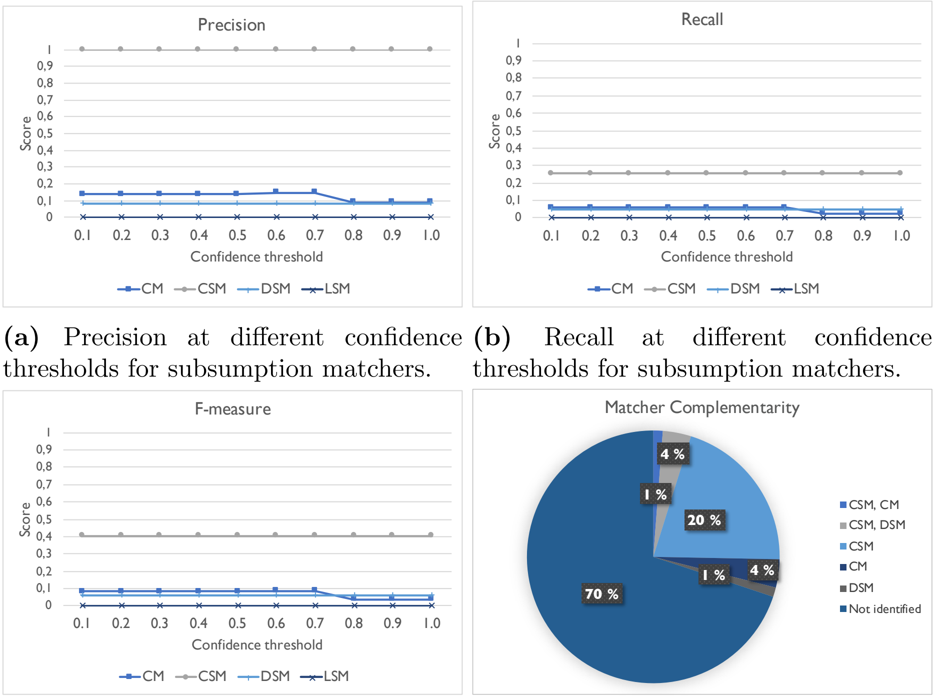
(d) Complementarity of the different equivalence matchers.

Figure 6.2: Evaluation scores for different equivalence matchers at different confidence thresholds.

Evaluation of Individual Subsumption Matchers

Figure 6.3 presents the precision, recall and F-measure scores of the subsumption matchers in the ATM dataset. The scoring functions of these matchers are, contrary to the equivalence matchers, based on boolean principles - a candidate relation is either a subsumption relation, and given a high confidence (0.75 or 1.0), or not given any confidence at all (0) - see Section 5.3 for details.

The reference alignment includes 83 relations in total, and only the Context Subsumption Matcher (CSM), Compound Matcher (CM) and Definition Subsumption Matcher (DSM) are able to identify any correct relations in this dataset. CSM produces an alignment containing 21 relations and manages to avoid any false positive ones. In terms of F-measure this is also the subsumption matcher that obtains the highest F-measure (0.4) as seen in Figure 6.3(c).



(a) Precision at different confidence thresholds for subsumption matchers.

(b) Recall at different confidence thresholds for subsumption matchers.

(c) F-measure at different confidence thresholds for subsumption matchers.

(d) Complementarity of the different subsumption matchers.

Figure 6.3: Evaluation scores for different subsumption matchers at different confidence thresholds

CM returns an alignment holding 34 relations, of which 4 are correct. When examining the false positives, most of them are based on the assumption that if two class names are syntactically equal their semantics are the same as well. However, in this dataset there are several occurrences of two classes sharing similar lexical properties, but where there is a more or less subtle deviation in their intended meaning. As described in the dataset summary (Section 6.2.1), 40 percent of the relations considered a "light match" consisted of class names with words in common. Furthermore, several of the relations returned by CM are in fact equivalence relations even if they fall under the compound head pattern applied by this matcher. Examples of this are $AircraftCapacity = Capacity$ and $AircraftEngine = Engine$.

Examining the alignment produced by the Definition Subsumption Matcher (DSM), it identifies 4 correct relations, of which one is not identified by any other subsumption matcher. Several of the false positive relations intuitively seemed correct, such as *MeteorologicalCondition* $>$ *Wind* and *SkyCondition* $>$ *Cloud*, however they are not present in the reference alignment. This is a good example of a conceptualisation mismatch (see Section 2.3) caused by different scope. *MeteorologicalCondition* and *WeatherPhenomenon*, the parent class of *Wind*, have different scope in the sense that *Meteorological* $-$ *Condition* has a focus on the meteorological conditions at an airport, while *WeatherPhenomenon* considers the meteorological conditions of the whole airspace (and includes the subclass *VolcanicAshCloud*, among others).

Around 25 percent of all relations in the subsumption reference alignment included ATM-specific acronyms in one or both of the class names. Without any structural criteria or natural language definition that could be used to deduce a relation involving these classes, the identification of these relations is a challenge.

6.2.3 Evaluation of Alignment Combination Methods

Table 6.2 shows the ontology profiling scores used to weight the different equivalence and subsumption matchers in the weighted variant of the alignment combination methods. Definition Coverage (DC) is marked in red, bold text since it is below the *matcher selection threshold* of 0.5. This results in that the matchers Definition Equivalence Matcher (DEM) and Definition Subsumption Matcher (DSM) will be omitted in the matcher ensemble when we evaluate the Profile Weight combination method with matcher selection.

Table 6.2: Ontology Profiling ATM Dataset

	Ontology Profiling Metric	**Measurement**
Terminological Analysis	Compound Fraction (CF)	0.93
	Corpus Coverage (CC)	0.84
	Definition Coverage (DC)	**0.29**
Structural Analysis	Property Fraction (PF)	0.78
	Structural Profile (SP)	0.58
Lexical Analysis	Lexical Coverage (LC)	0.72

Figure 6.4, Figure 6.5 and Figure 6.6 show evaluation scores in terms of precision, recall and F-measure (respectively) in the ATM dataset. The Profile Weight configurations do not rely on any specified threshold (all relations in the final alignment are considered equally valid), hence the evaluation scores are constant across all confidence thresholds.

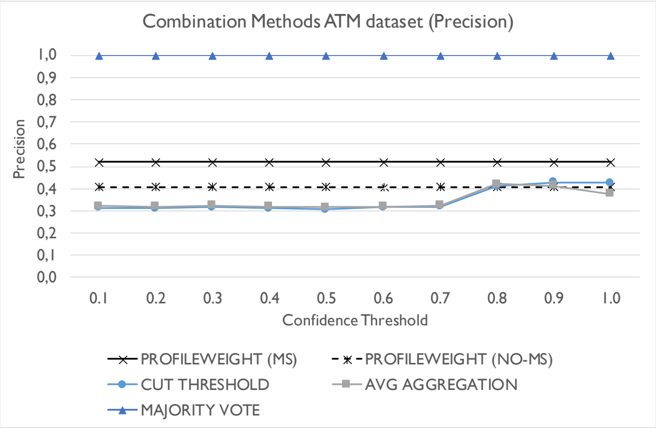


Figure 6.4: Precision of Alignment Combination Methods in the ATM dataset.

Profile Weight with matcher selection obtains the highest F-measure score in this dataset, with an F-measure of 0.41. The second highest F-measure is obtained by Profile Weight without matcher selection (0.38), followed by Average Aggregation (0.35 at confidence threshold 0.8) and Cut Threshold (0.34 at confidence thresholds ≥ 0.8). When combining the alignment using Majority Vote, 9 relations are preserved from the individual alignments, of which 5 are equivalence relations and 4 are subsumption relations. All of these 9 are correct, something which yields a precision of 1.0, but with a

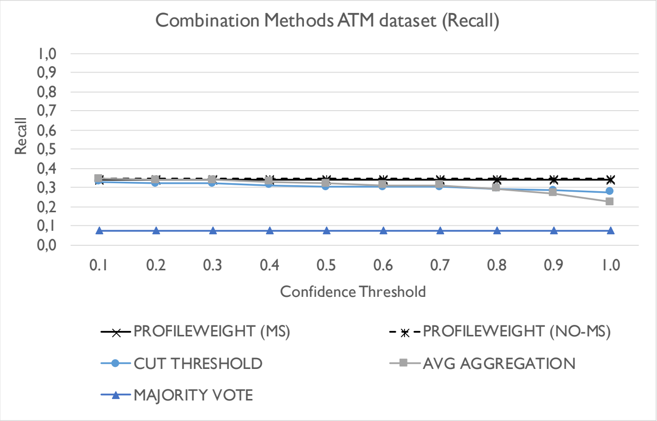


Figure 6.5: Recall of Alignment Combination Methods in the ATM dataset.

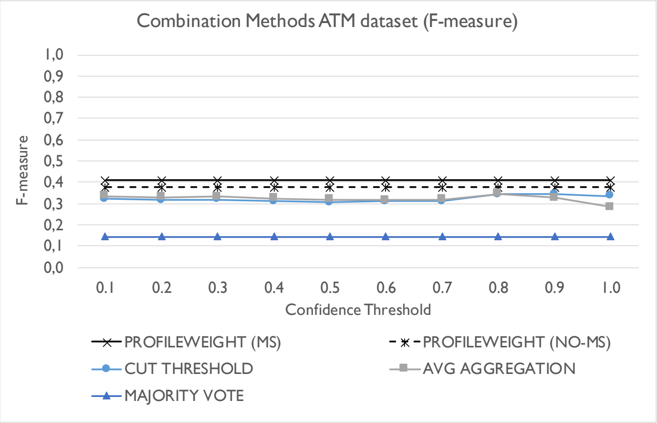
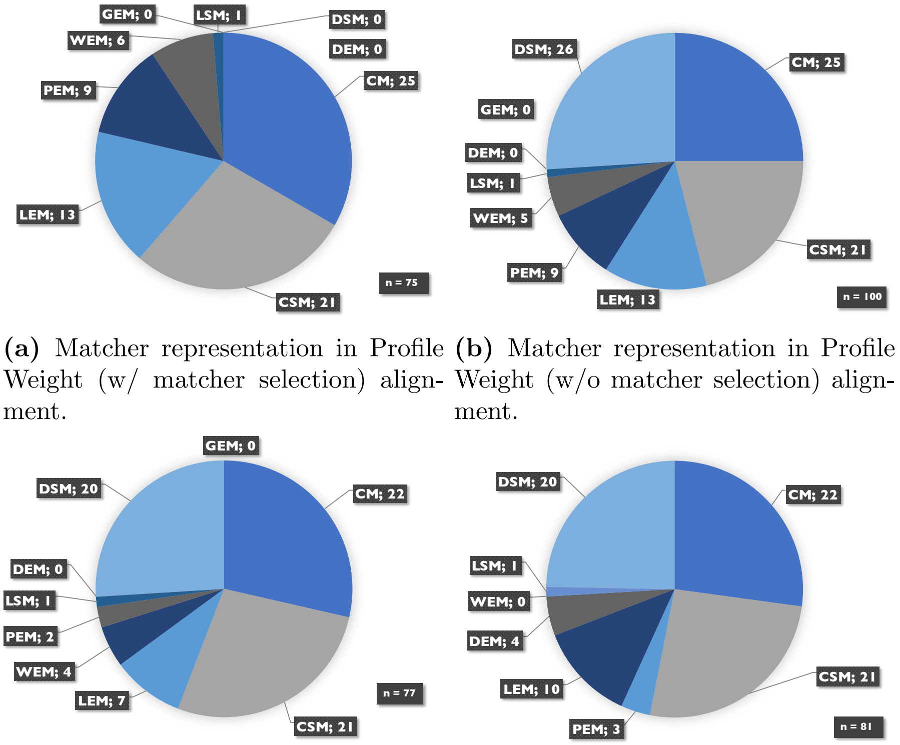


Figure 6.6: F-measure of Alignment Combination Methods in the ATM dataset.

low recall of 0.08, the F-measure obtained by this combination method is 0.15. Profile Weight manages to balance the trade-off between precision and recall better than the other three combination methods although at some thresholds they obtain either higher precision (at higher thresholds) or the same recall (at lower thresholds) as Profile Weight.

Furthermore, as illustrated in Figure 6.7, the alignment produced by the Profile Weight methods includes more relations from the best performing equivalence matchers Lexical Equivalence Matcher and Word Embedding Matcher than the Cut Threshold method at its best performing confidence threshold (≥ 0.8). For the subsumption relations, and when examining the alignments in detail, Profile Weight excludes more of the false positive relations from the Compound Matcher while preserving the same true positive relations from the Context Subsumption Matcher as the other two combination methods.

The Profile Weight alignment with matcher selection obtains a higher precision than without matcher selection (0.52 and 0.41 respectively), partly because many of the false positive relations brought in by the Definition Subsumption Matcher are omitted.



(a) Matcher representation in Profile Weight (w/ matcher selection) alignment.

(b) Matcher representation in Profile Weight (w/o matcher selection) alignment.

(c) Matcher representation in Cut Threshold alignment.

(d) Matcher representation in Average Aggregation alignment.

Figure 6.7: Illustrations showing how relations from individual matchers are represented in the combined alignments.

An ensemble of individual matchers is used with the intuition that they collectively produce a higher quality alignment than what they are capable of on their own. The F-measure of the alignment combination methods is therefore compared against the alignments produced by the best individual equivalence and subsumption matchers. Figure 6.8 and Figure 6.9 show the results from the comparison. The best individual equivalence matcher in this dataset is the Lexical Equivalence Matcher which at confidence threshold 0.9 obtained an F-measure of 0.44. When applied on the alignments produced by the individual equivalence matchers, the Profile Weight combination method (both with and without matcher selection) obtains an F-measure of 0.52, so this combination method enhances the individual alignments. When combining the individual alignments using Average Aggregation the highest F-measure is 0.48. For the Cut Threshold combination the highest F-measure achieved is 0.43 at confidence threshold 0.1 and 0.3. The Majority Vote combination method returned an equivalence alignment with an F-measure of 0.27 at all confidence thresholds.

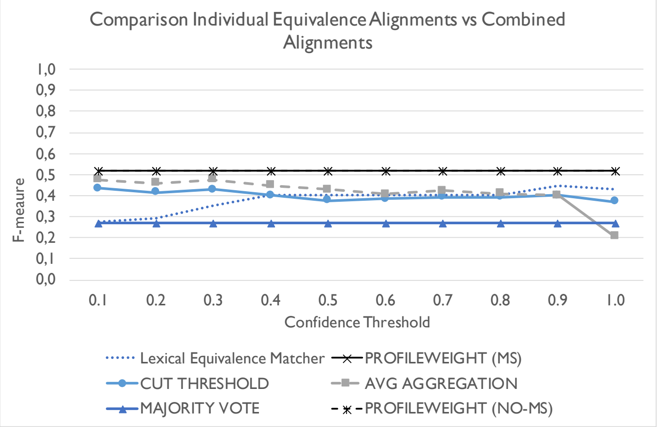


Figure 6.8: Comparison of Best Individual Equivalence Alignment and Combined Alignments in the ATM dataset.

The best individual subsumption matcher in this dataset is the Context Subsumption Matcher (CSM) obtaining an F-measure of 0.4. The Profile Weight combination method, when applied on the individual subsumption matchers, achieves an F-measure of 0.37 when the matcher selection is applied (precision 0.51 and recall 0.29), and an F-measure of 0.32 without matcher selection. The Cut Threshold method returns its best F-measure of 0.33 at confidence thresholds 0.8 and upwards, while Average Aggregation returns an alignment with an F-measure of 0.33 at confidence threshold 0.8. Here, the subsumption alignment produced by the Majority Vote method obtained an F-measure of 0.09 across all confidence thresholds. So for the subsumption matching we experience a reduction of alignment quality when applying the combination methods in this dataset.

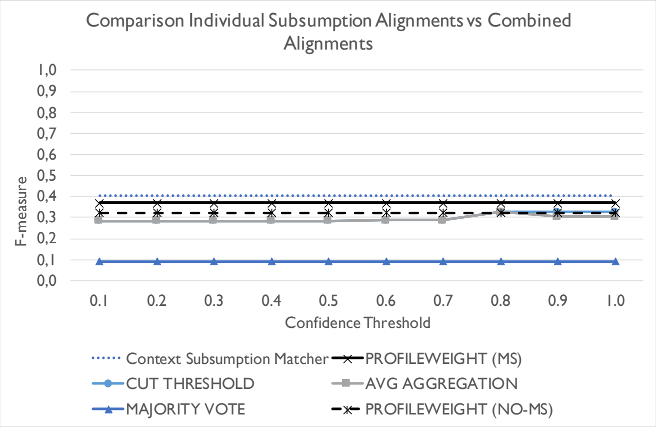


Figure 6.9: Comparison of Best Individual Subsumption Alignment and Combined Alignments in the ATM dataset.

Figure 6.10, Figure 6.11 and Figure 6.12 show the comparison with the other existing matching systems that return alignments holding both equivalence and subsumption relations. BLOOMS with its WordNet configuration includes only true positive relations in its alignment, but scores low on recall. STROMA achieves 100 percent precision at confidence threshold 0.9. All true positives identified by these two systems are equivalence relations. S-Match suffer from very low precision due to extensive alignments with many false positives. S-Match's alignment includes 453 relations, of which only two are correct. BLOOMS (WIKI) returns an alignment composed of 10 relations, of which 3 are correct.

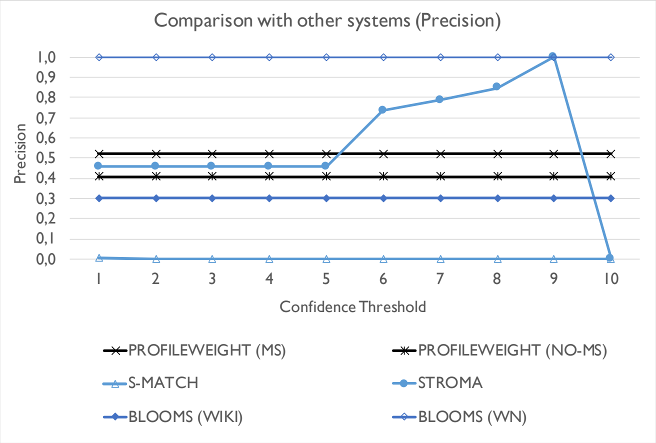


Figure 6.10: Comparing systems on precision in the ATM dataset.

The highest F-measure is obtained by Profile Weight with matcher selection, followed by Profile Weight without matcher selection.

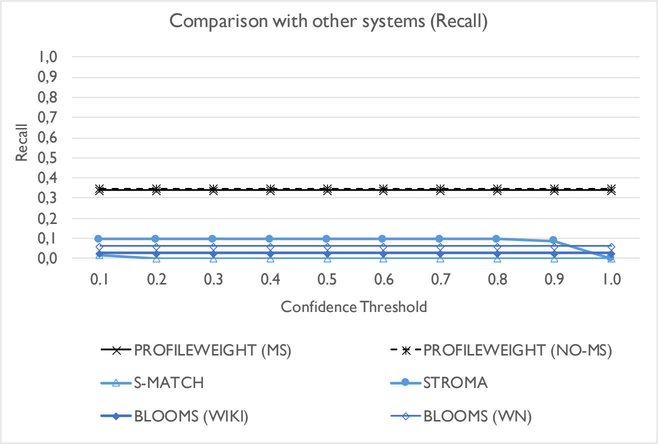


Figure 6.11: Comparing systems on recall in the ATM dataset.

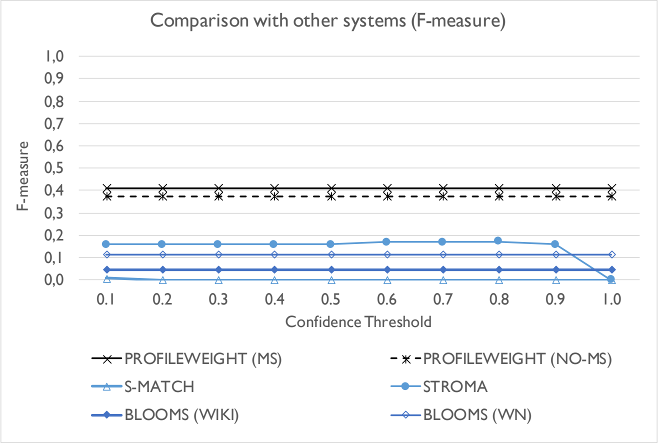


Figure 6.12: Comparing systems on F-measure in the ATM dataset.

Figures 6.13, 6.14 and 6.15 show the evaluation scores when considering semantic relations inferred from a reasoner. Here, semantic precision, recall and F-measure are computed for alignments extracted from an initial alignment at each confidence interval. The systems that initially only return equivalence relations in their alignments are presented with dotted lines in the chart.

In general the scores are higher here than when using traditional precision and recall measures as presented in the previous evaluation. LogMap and BLOOMS (WIKI) obtain the highest semantic precision, while Profile Weight (including both equivalence and subsumption relations) achieve the highest recall.

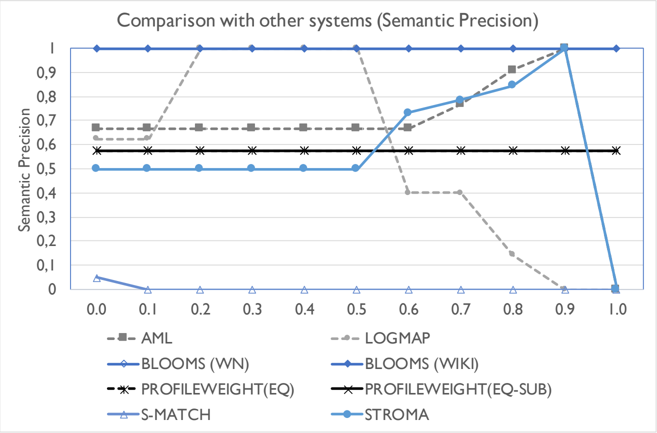


Figure 6.13: Comparing systems on semantic precision in the ATM dataset.

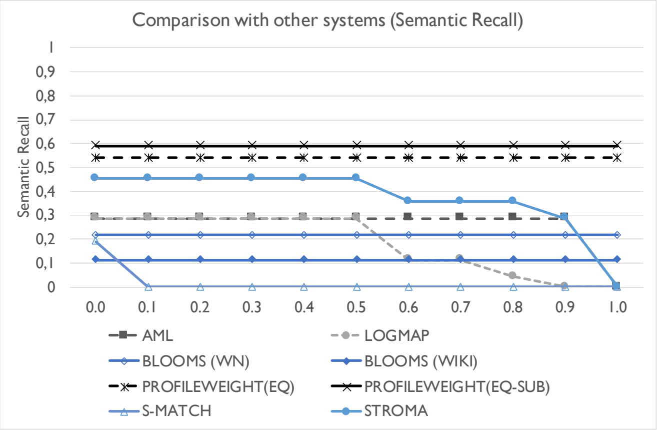


Figure 6.14: Comparing systems on semantic recall in the ATM dataset.

Looking at the F-measure scores in Figure 6.15, we see that the Profile Weight variant which includes both equivalence and subsumption relations in its alignment obtains the highest score. If we compare with the initial alignment produced by Profile Weight (as used in the previous evaluation), 28 new correct relations were inferred, contributing to a significant increase in recall. The Profile Weight variant that only includes equivalence relations does also perform relatively well, and identifies more correct relations than the other systems that rely on inferred subsumption relations.

AgreementMakerLight, STROMA and LogMap identified 3 relations that were not captured by Profile Weight. These three relations were: SID = StandardInstrumentDeparture, SID < Procedure, and SIDSTAR > StandardInstrumentDeparture. The reason for this is that SID = StandardInstrumentDeparture was identified by AgreementMakerLight and LogMap when computing their equivalence alignments, while this relation was not identified by any of the equivalence matchers proposed in this work. The other two relations were inferred from SID = StandardInstrumentDeparture. As the equivalence alignment from AgreementMakerLight was used to bootstrap the subsumption matching in STROMA, these three relations were also included in the STROMA alignment in this evaluation.

43 relations (16 equivalence and 27 subsumption relations) included in the reference alignment were not identified by any of the systems involved.

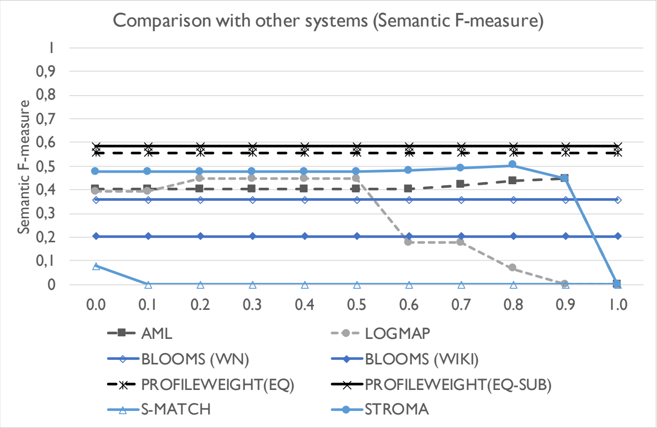


Figure 6.15: Comparing systems on semantic F-measure in the ATM dataset.

6.2.4 Evaluation of Mismatch Detection Strategies

The mismatch detection strategies are described in Section 5.5. When running the mismatch detection on the Profile Weight combination method, this increased the F-measure of the equivalence alignment from 0.32 to 0.52. The precision increased from 0.25 to 0.58 while the recall remained stable at 0.47.

The Domain Mismatch Detection strategy made the strongest contribution as it filtered out 34 relations from the initial alignment, all of which were false positives. The Concept Scope Mismatch Detection strategy filtered out one relation, however this was incorrectly filtered out.

6.3 Dataset 2 - Cross Domain

This dataset involves the two ontologies BIBFRAME[9] and Schema.org[10]. These ontologies are of different generality level, BIBFRAME is a domain ontology for bibliographic information, while Schema.org is a general purpose ontology, typically used as a vocabulary for web resources. The reference alignments in this dataset are developed by the author. The process of developing the reference alignments is conducted as follows:

1. Generate a candidate equivalence reference alignment by producing alignments using the three ontology matching systems AgreementMakerLight [39], LogMap [78] and YAM++ [102].

2. Correct and extend the candidate reference alignment by manually inspecting the BIBFRAME and Schema.org ontologies. This inspection included an analysis of classes, associated properties as well as neighbouring classes.

3. Transform the resulting reference alignment into the Alignment Format [11].

4. Create a subsumption reference alignment by following the approach described in Section 6.2.1.

6.3.1 Dataset Summary

Table 6.3 describes some statistics of the two ontologies in this dataset. Schema.org is larger than BIBFRAME in terms of classes, object properties and data properties, and is more general in scope than its bibliographic counterpart. In both ontologies the entities are annotated with natural language definitions. Compared to the ontologies in the ATM dataset, the terminology used in these two ontologies is more general and in the equivalence reference alignment 11 of the 16 relations are exact string matches. In the remaining relations both the source and target concepts have words in

[9] http://id.loc.gov/ontologies/bibframe.html

[10] https://schema.org/docs/schemaorg.owl

[11] This step included removing some of the meta-data tags from the alignment produced by AgreementMakerLight as well as converting the skos:exactMatch relation type produced by YAM++ to "=".

common (e.g. *MusicFormat* = *MusicReleaseFormatType*). The subsumption reference alignment contains a large number of relations where a single class in one ontology is related to many classes in the second ontology. One example is how the class *Event* in BIBFRAME subsumes 19 specific events (e.g. *TheaterEvent* and *Festival*) described in Schema.org. In the majority of cases the compound pattern strikes in, meaning that the subsuming class is represented in the compound head of the subsumed class, e.g. *Organization* > *SportsOrganization*. Contrary to the ATM dataset, most class names consist of commonly used words that are covered in lexical resources such as WordNet or BabelNet. However, the representation of compounds is high, so some pre-processing of the words is required.

Table 6.3: Ontology statistics for the Cross Domain Dataset

	Classes	Object Properties	Data Properties	Axioms
BIBFRAME	188	132	63	2199
Schema.org	670	916	491	9171

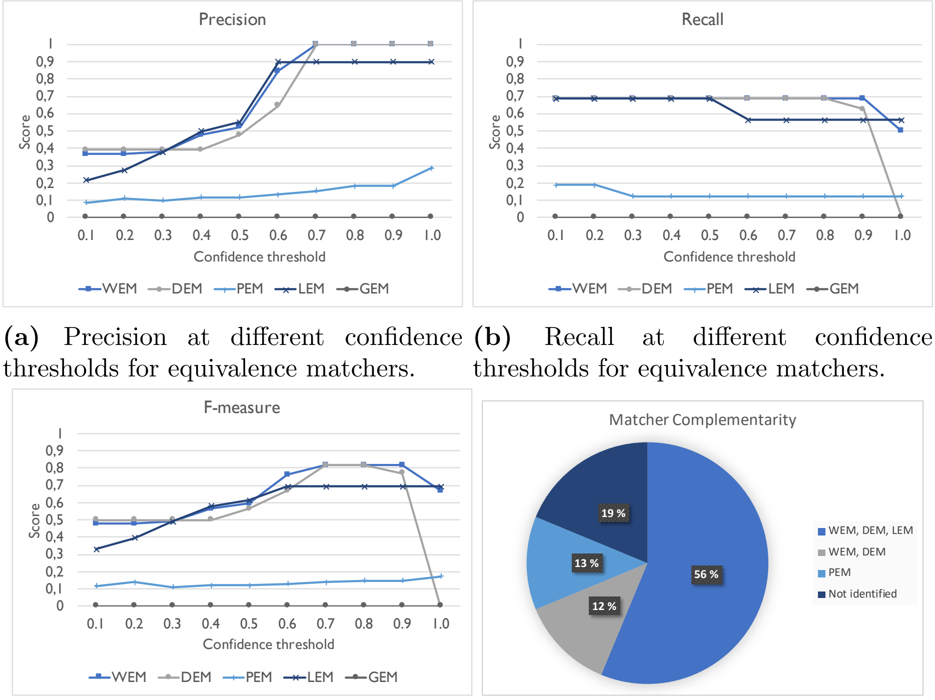
6.3.2 Evaluation of Individual Matchers

Evaluation of Individual Equivalence Matchers

Figure 6.16 shows the precision, recall and F-measure scores for the equivalence matchers in this dataset. Overall, the best performing matchers are the Word Embedding Matcher (WEM) and the Definition Equivalence Matcher (DEM) which both obtain an F-measure of around 0.82 at confidence levels 0.7 and 0.8. Both matchers have a perfect precision at these thresholds with no false positives. The relations in the reference alignment these two matchers did not identify do all have concepts with words in common (e.g. *BookFormat* = *BookFormatType* or *Audio* = *AudioObject*). The Lexical Equivalence Matcher (LEM) obtains an F-measure of around 0.7 at confidence threshold 0.6 and higher. This matcher discovered many of the same relations as WEM and DEM. The Property Equivalence Matcher (PEM) achieves its best F-measure (0.17) at confidence value 1.0. PEM discovers two correct relations that are missed by the other matchers and thus makes a contribution to the complementarity of the matchers. The Graph Equivalence Matcher (GEM) does not identify any true positive equivalence relations in this dataset.

Looking at the matcher complementary chart in Figure 6.16(d) we see that 81 % (13 of 16) of all relations were identified by one or more matchers.

The three that were not identified all have naming patterns that suggest another relation type than equivalence. *Audio* = *AudioObject* and *Media* = *MediaObject* suggest meronomy whereas *IntendedAudience* = *Audience* suggest subsumption as per the analysis described in Section 5.5. Further we see that 56 % of all relations in the reference alignment are identified by 3 (i.e. the majority) of the matchers.



(a) Precision at different confidence thresholds for equivalence matchers.

(b) Recall at different confidence thresholds for equivalence matchers.

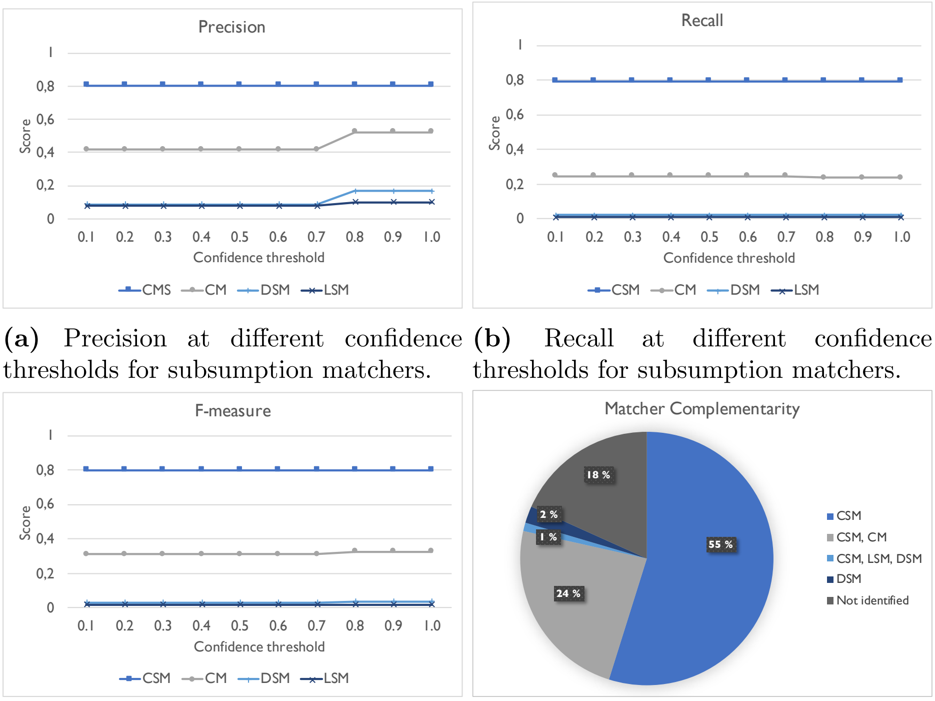
(c) F-measure at different confidence thresholds for equivalence matchers.

(d) Complementarity of the different equivalence matchers.

Figure 6.16: Evaluation scores for different equivalence matchers at different confidence thresholds in the Cross-Domain dataset.

Evaluation of Individual Subsumption Matchers

Figure 6.17 shows the evaluation scores for the individual subsumption matchers for this dataset. The Context Subsumption Matcher (CSM) which is based on the contextual similarity with respect to similar super- and subclasses perform the best also in this dataset, obtaining an F-measure of 0.8 at all confidence thresholds. As mentioned in the summary of the dataset in Section 6.3.1, the reference alignment contains a large number of relations where one class in the source ontology subsumes many classes in the target ontology if it is equivalent to the superclass of these classes in the target ontology, so this dataset fits CSM well.



(a) Precision at different confidence thresholds for subsumption matchers.

(b) Recall at different confidence thresholds for subsumption matchers.

(c) F-measure at different confidence thresholds for subsumption matchers.

(d) Complementarity of the different subsumption matchers.

Figure 6.17: Evaluation scores for different subsumption matchers at different confidence thresholds in the Cross-Domain dataset.

The second best matcher is the Compound Matcher (CM) that for the confidence thresholds 0.8 and above obtains an F-measure of around 0.33. CM produces an alignment consisting of 42 relations, of which 22 are correct. All of the correct relations identified by CM are also identified by the sub- and superclass pattern used by CSM, so in terms of contribution to the matcher complementarity CM is not very useful in this dataset. The false positive relations identified by CM are a result of identification of transitive subsumption relationships (e.g. CM suggest that the source class *Organization* subsumes the target class *MedicalOrganization*, but the taxonomy on the target side is *Organization* > *LocalBusiness* > *MedicalOrganization*), differing scope of classes (e.g. CM suggests that *Object* > *AudioObject* whereas *Object* has a disjoint sibling *Audio* which is equivalent to *AudioObject*), and simply that the syntactics does not match the semantics (e.g. CM suggests that *IntendedAudience* < *Audience* but according to definition and context these are equivalent classes).

The Definition Subsumption Matcher (DSM) identifies two correct subsumption relations, none of which are identified by any other matcher. These are *Agent* > *Person* and *Agent* > *Organization*. Both these relations are captured by the "Such as..." lexico-syntactic pattern. Many of the false positive relations produced by DSM suggest some other type of semantic relatedness between the concepts than subsumption. For example, the relation *Publication* - *PublicationIssue* suggests a meronymic relation (as in a publication has an issue), and *Place* - *PropertyValue* (here, *PropertyValue* is a structured value at the same level as a quantitative value or a price specification).

The Lexical Subsumption Matcher (LSM) identifies one correct relation which is also identified by CSM, namely *Organisation* > *NGO* (acronym for Non-Governmental Organization). The false positive relations suggested by LSM are mostly due to misinterpretation of the semantics of a class. For example, the concept *Work* in BIBFRAME is defined as a "*Resource reflecting a conceptual essence of a cataloging resource*", whereas in WordNet, which the LSM uses as a lexical resource, *Work* is defined as an "*Activity directed toward making or doing something*". On this basis LSM produces the subsumption relations *Work* > *Service* and *Work* > *Action* where *Service* and *Action* are hyponyms of *Work* in WordNet - which both are wrong in this context.

6.3.3 Evaluation of Alignment Combination Methods

Table 6.4 shows the scores from the ontology profiling of the BIBFRAME and Schema.org ontologies.

The most noticeable differences are that compared with the ATM dataset, the Compound Fraction is lower in this dataset (0.91 in the ATM dataset), the Property Fraction is lower (0.78 in the ATM dataset), and the Structure Profile is significantly higher (0.58 in the ATM dataset). The Corpus Coverage, Definition Coverage, and Lexical Coverage scores are more or less the same as in the ATM dataset (respectively 0.84, 0.29, and 0.72 in the ATM dataset).

Since the Definition Coverage (DC) and the Property Fraction (PF) scores are below the threshold used for determining which matchers should take part in the matcher ensemble, these are marked in red, bold text in Table 6.4. The consequence of this is that the Definition Equivalence Matcher (DEM), the Definition Subsumption Matcher (DSM) and the Property Equivalence Matcher (PEM) are omitted in one of the Profile Weight combination method configurations.

Table 6.4: Ontology Profiling Cross-domain Dataset

	Ontology Profiling Metric	**Measurement**
Terminological Analysis	Compound Fraction (CF)	0.65
	Corpus Coverage (CC)	0.83
	Definition Coverage (DC)	**0.27**
Structural Analysis	**Property Fraction (PF)**	**0.41**
	Structural Profile (SP)	0.91
Lexical Analysis	Lexical Coverage (LC)	0.75

Figures 6.18 (precision), 6.19 (recall) and 6.20 (F-measure) show the performance of the different alignment combination methods in the ATM dataset.

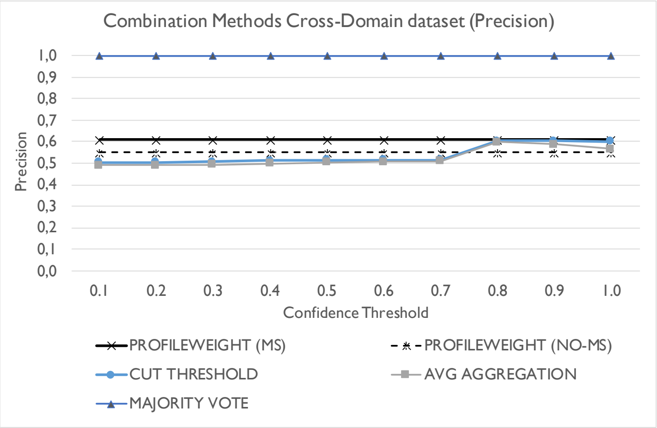


Figure 6.18: Precision of Alignment Combination Methods in the Cross-domain dataset.

As illustrated in Figure 6.20, the Profile Weight method obtains an F-measure that is higher than the other three combination methods at all thresholds lower than 0.8 – both when applying matcher selection and not. This is primarily contributed by a lower number of false positives, hence a higher precision, in the Profile Weight alignments, with the exception of

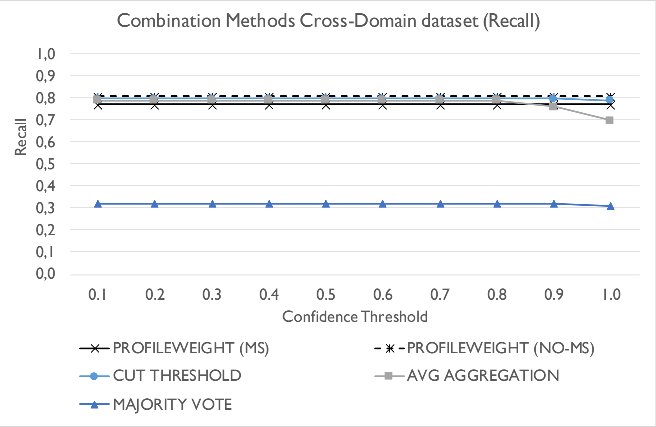


Figure 6.19: Recall of Alignment Combination Methods in the Cross-domain dataset.

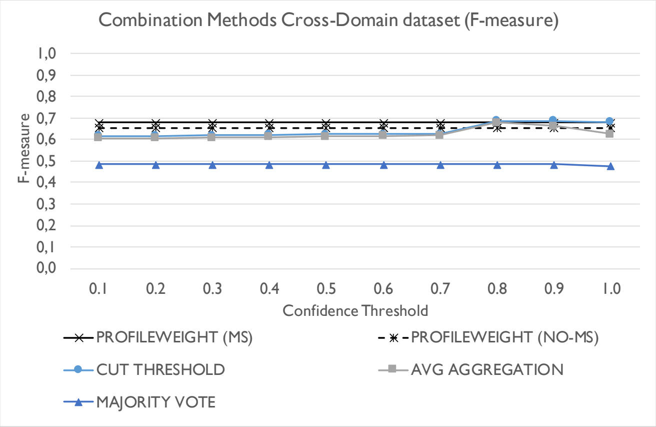
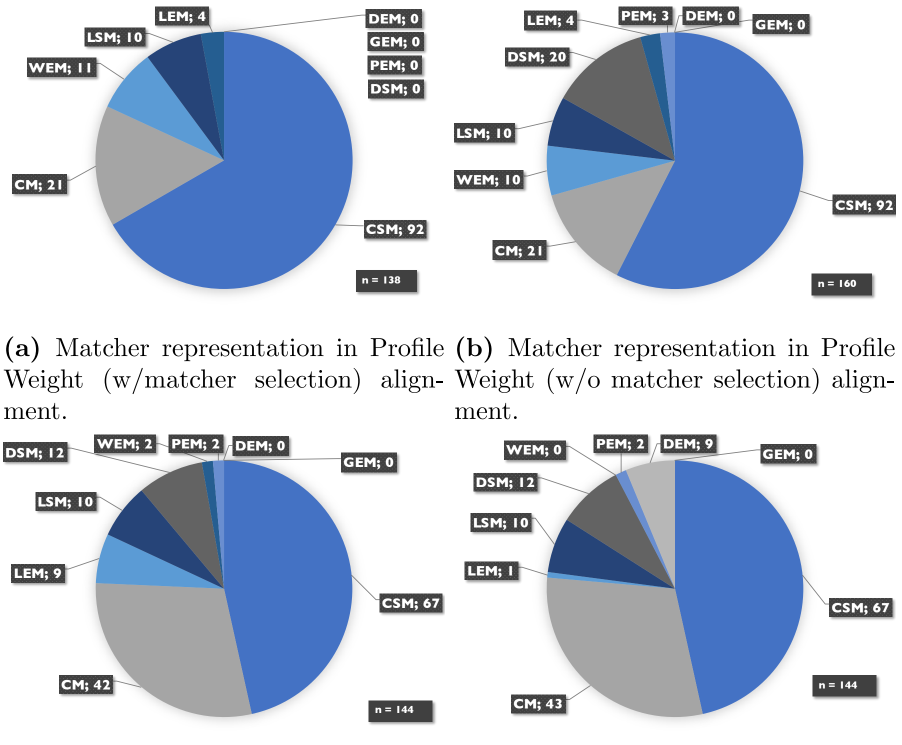


Figure 6.20: F-measure of Alignment Combination Methods in the Cross-domain dataset.

Majority Vote, which has a perfect precision, but has a much lower recall than the other methods Cut Threshold and Average Aggregation obtain a slightly higher F-measure at confidence threshold 0.8, which is maintained by the Cut Threshold method at confidence thresholds 0.9 and 1.0.

Figure 6.21 illustrates how the relations from the individual equivalence and subsumption matchers are represented in the alignment produced by Profile Weight and in the alignments produced by the other methods at their optimal confidence thresholds (0.8 for all of them). The alignment produced by Profile Weight includes many relations that have low confidence values, most of which are false positive relations, especially when not using the matcher selection. For example, in this alignment there are 20 relations produced by the Definition Subsumption Matcher (DSM), of which only 2 are correct. When using the matcher selection, this is corrected since due to low Definition Coverage (see Section 5.1) the Definition Subsumption Matcher is omitted in this configuration. Cut Threshold and Average Aggregation disregard the DSM relations with the lowest confidence value, and only accept 12 relations from this matcher into the final alignment, and the two correct relations are preserved. Furthermore, there are 21 relations in the Profile Weight alignment produced by the Compound Matcher (CM), and all are incorrect. Cut Threshold and Average Aggregation include 43 and 42 relations from the Compound Matcher (CM) respectively and in both alignments 22 of them are true positives.



(a) Matcher representation in Profile Weight (w/matcher selection) alignment.

(b) Matcher representation in Profile Weight (w/o matcher selection) alignment.

(c) Matcher representation in Cut Threshold alignment.

(d) Matcher representation in Average Aggregation alignment.

Figure 6.21: Illustrations showing how relations from individual matchers are represented in the combined alignments.

When comparing the performance of the combination methods with the individual matcher, the best performing equivalence matchers were Word Embedding Matcher (WEM) and Definition Equivalence Matcher (DEM), both achieving an F-measure of 0.81 at confidence thresholds 0.7 and 0.8. When isolating the equivalence relations, Profile Weight obtains an F-measure of 0.65 when matcher selection is applied, and 0.73 when it is not, as shown in Figure 6.22. The best Cut Threshold alignments returns an F-measure of 0.83 at confidence thresholds 0.5-0.9 for the equivalence alignment. Average Aggregation produces a combined equivalence alignment that at confidence thresholds 0.7 and 0.8 get an F-measure of 0.79. Majority Vote produces an F-measure of 0.81 at confidence thresholds range 0.1-0.9. Hence, several of the combination methods achieve a higher score than Profile Weight with matcher selection when only equivalence relations are considered.

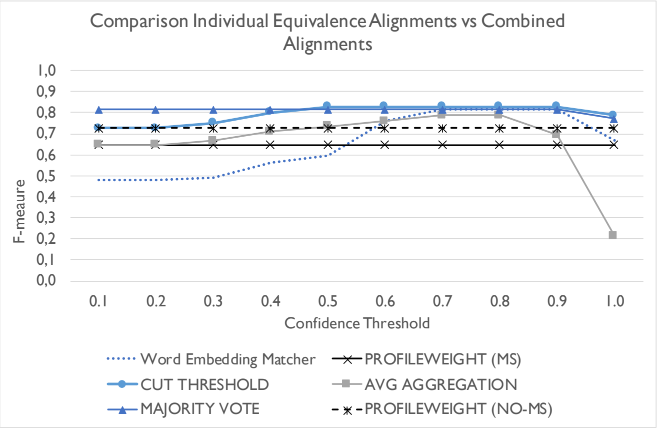


Figure 6.22: Comparison of Best Individual Equivalence Alignment and Combined Alignments in the Cross-domain dataset.

For the subsumption matching, the best individual matcher was Context Subsumption Matcher (CSM) which at all confidence thresholds obtained an F-measure of 0.8. As Figure 6.23 illustrates, when combining all alignments from the individual subsumption matchers, Profile Weight with matcher selection gets an F-measure of 0.69, while Profile Weight without matcher selection yields an F-measure of 0.64. Average Aggregation produces a combined subsumption alignment that at confidence thresholds 0.8 gets an F-measure of 0.68. Cut Threshold returns a combined subsumption alignment obtaining an F-measure of 0.67 at confidence thresholds 0.8, 0.9 and 1.0. The Majority Vote alignment obtains an F-measure of 0.41 across all confidence thresholds.

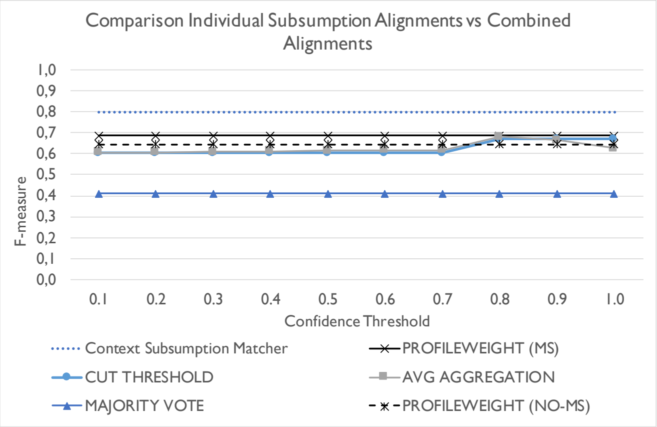


Figure 6.23: Comparison of Best Individual Subsumption Alignment and Combined Alignments in the Cross-domain dataset.

We see the same patterns in this dataset when comparing the best individual alignments with the combined alignments. For the equivalence matching the combination of the alignments produced by the individual matchers enhances the quality in terms of F-measure, whereas for the subsumption matching the alignments produced by the best individual matchers achieve higher F-measure scores.

Figure 6.24, Figure 6.25 and Figure 6.26 show the comparison on precision, recall and F-measure respectively with other systems producing both equivalence and subsumption relations in the cross-domain dataset.

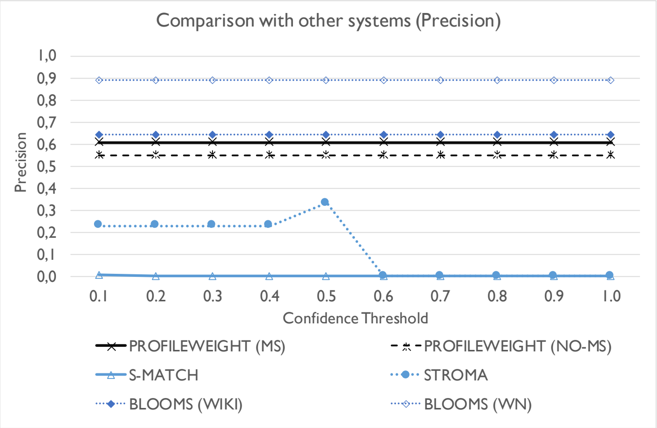


Figure 6.24: Comparing systems on precision in the Cross-domain dataset.

As in the ATM dataset, Profile Weight (Profile Weight EQ-SUB) obtains higher F-measure scores than the other systems producing both equivalence and subsumption relations. Both BLOOMS configurations obtain a higher precision than Profile Weight, but experience a low recall. S-Match's optimal alignment (at confidence threshold 0.0) is very extensive also in this dataset. Out of a total of 619 returned relations there are only 6 true positives.

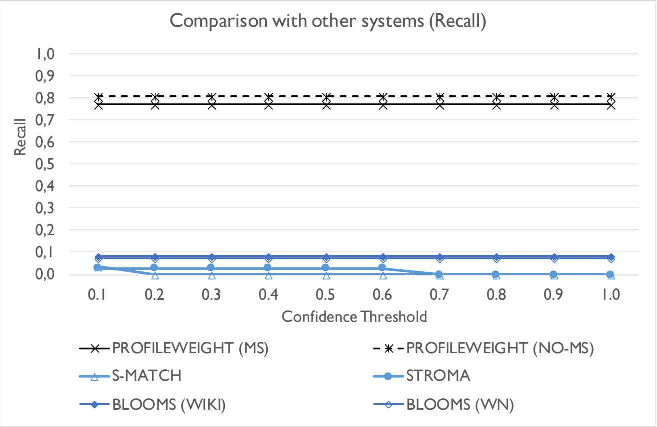


Figure 6.25: Comparing systems on recall in the Cross-domain dataset.

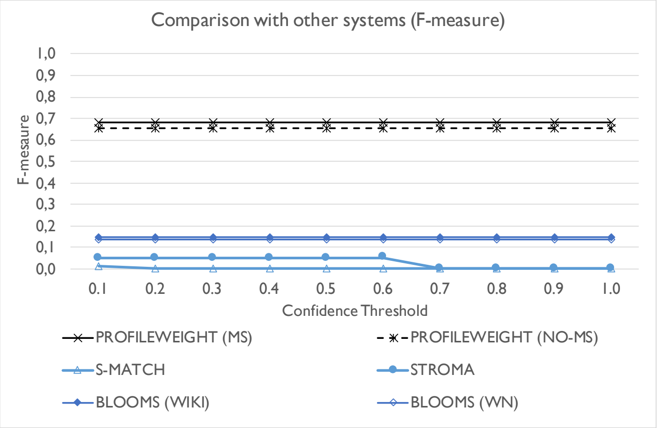


Figure 6.26: Comparing systems on F-measure in the Cross-domain dataset.

Figure 6.27, Figure 6.28 and Figure 6.29 illustrate the results from the second evaluation in this dataset where semantic precision and recall evaluation measures are applied. 18 (out of 109) relations in the reference alignment were not found by any of the systems. 14 of these relations were subsumption relations, while 4 were equivalence relations.

AgreementMakerLight performs best in this evaluation with an F-measure of 0.82 at confidence thresholds 0.0-0.7. Profile Weight obtains the second highest F-measure of 0.76 (with a semantic precision of 0.72 and a semantic recall of 0.79), with BLOOMS (WIKI) close behind at 0.74.

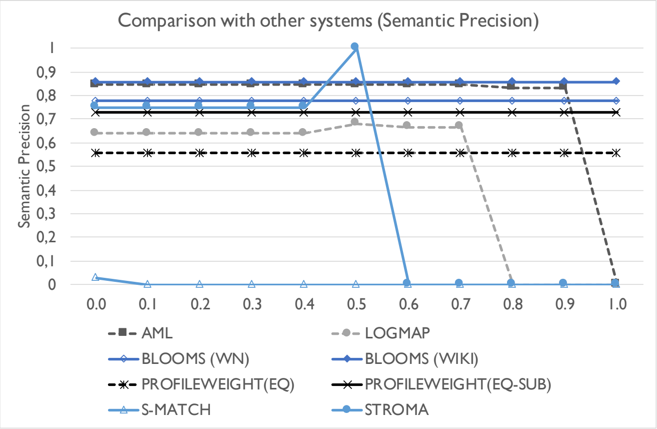


Figure 6.27: Comparing systems on Semantic Precision in the Cross-domain dataset.

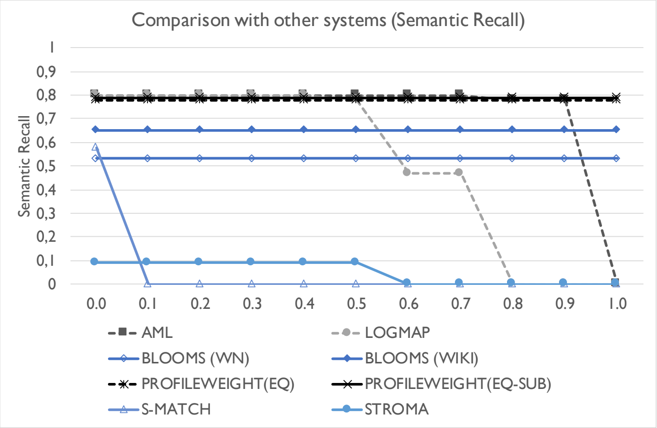


Figure 6.28: Comparing systems on Semantic Recall in the Cross-domain dataset.

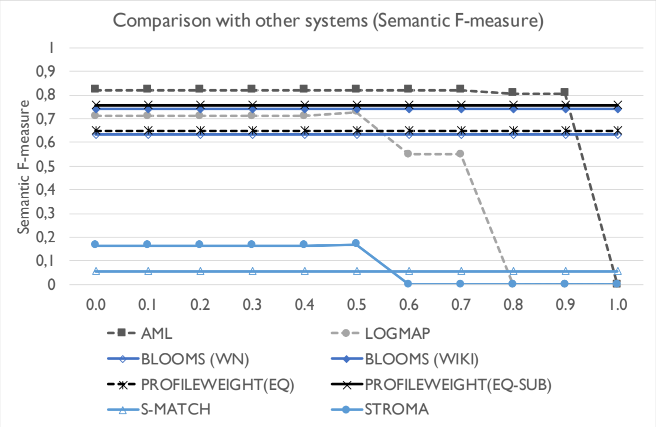


Figure 6.29: Comparing systems on Semantic F-measure in the Cross-domain dataset.

Compared to the initial alignment holding equivalence and subsumption relations produced by Profile Weight, 3 new correct relations are inferred (Agent > Person, Agent > Organization and IntendedAudience < Intangible). When comparing the inferred Profile Weight alignment (with matcher selection) with the inferred alignment from AgreementMakerLight, we see that there are 3 correct relations not included in the Profile Weight alignment: Person = Person, BookFormat = BookFormatType and BookFormat < Enumeration. In the Profile Weight alignment Person = Person was proposed by the Word Embedding Matcher with a confidence of 0.801, but was overridden when merging the equivalence and subsumption alignments by Person > Person from the Context Subsumption Matcher with a confidence of 0.809. BookFormat = BookFormatType was suggested by the Property Matcher, however when applying the matcher selection this matcher was not included in the ensemble since the Property Fraction metric was too low. BookFormat < Enumeration was not identified by any of the subsumption matchers.

6.3.4 Evaluation of Mismatch Detection Strategies

The mismatch detection increased the F-measure of the equivalence alignment produced by the Profile Weight method (with matcher selection) from 0.30 to 0.65 thanks to the Domain Mismatch Detection strategy. This strategy filtered out 41 relations and all of them were false positive relations, increasing the precision from 0.19 to 0.67. The recall was not affected by the Domain Mismatch Detection strategy. The Concept Scope Mismatch Detection did not filter out any relation from the initial alignment.

6.4 Dataset 3 - OAEI

This dataset is based on the 'Oriented Matching' track from the Ontology Alignment Evaluation Initiative in 2011. This is the latest OAEI campaign containing subsumption relations in the reference alignment[12].

6.4.1 Dataset Summary

This dataset consists of 4 ontologies that all describe conference organisation. These ontologies are distributed into 6 sub-datasets representing possible permutations from these four ontologies. For each sub-dataset a combined equivalence and subsumption reference alignment is constructed. As seen in the ontology statistics presented in Table 6.5, the ontologies in this dataset are smaller than in the two previous datasets. Most classes are represented by non-technical and common terminology and in only one of the sub-datasets (301304) the classes include natural language definitions. Each individual ontology is slightly modified from sub-dataset to sub-dataset, as indicated by the number of classes listed for each ontology. For example, ontology 301 includes an additional class in sub-dataset 301302 compared to how this ontology is represented in sub-datasets 301303 and 301304.

Table 6.5: Ontology statistics for the OAEI 2011 Dataset

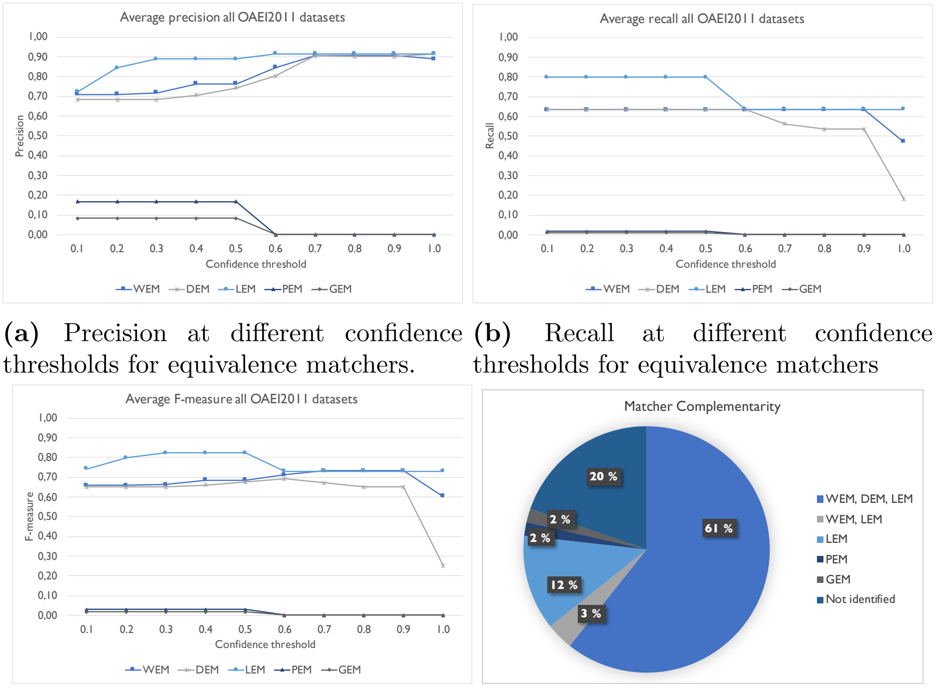
Sub-dataset	Classes	Object Properties	Data Properties	Annotation Properties	Axioms
301302-301	16	0	40	12	323
301302-302	16	6	25	2	169
301303-301	15	0	40	12	269
301303-303	54	72	0	2	569
301304-301	15	0	40	12	269
301304-304	36	40	11	10	437
302303-302	16	6	25	2	173
302303-303	53	72	0	2	561
302304-302	16	6	25	2	163
302304-304	35	40	11	10	431
303304-303	56	72	0	2	580
303304-304	41	40	11	10	467

[12] In OAEI 2009 there was also an Oriented Matching track that included subsumption relations in addition to equivalence relations in the reference alignments.

6.4.2 Evaluation of Individual Matchers

Evaluation of Individual Equivalence Matchers

Figure 6.30 shows evaluation scores obtained by averaging the results of the individual equivalence matchers in each dataset. As the figure shows there is a clear separation between the three best performing matchers Lexical Equivalence Matcher (LEM), Word Embedding Matcher (WEM) and Definition Equivalence Matcher (DEM), and the Property Equivalence Matcher (PEM) and Graph Equivalence Matcher (GEM). The overall best performing equivalence matcher is LEM which at confidence thresholds 0.2 - 0.5 achieves an F-measure of 0.82. This followed by WEM obtaining an F-measure of 0.73 at confidence threshold 0.7 and above. DEM achieves an F-measure of 0.69 (at confidence threshold 0.6).



(a) Precision at different confidence thresholds for equivalence matchers.

(b) Recall at different confidence thresholds for equivalence matchers

(c) F-measure at different confidence thresholds for equivalence matchers

(d) Complementarity of different equivalence matchers.

Figure 6.30: Average evaluation measures for individual equivalence matchers in the OAEI 2011 datasets

There is significant overlap between the alignments produced by LEM, WEM and DEM. Out of all true positive relations identified in all 6 sub-datasets, 76 % of them are identified by all three matchers as illustrated in Figure 6.30(d). When analysing the alignments in each sub-dataset individually, a recurring pattern is that WEM and DEM obtain the highest F-measure at high confidence thresholds (0.7 and higher) whereas LEM scores higher at lower thresholds (0.5 and below). This is also illustrated in the chart in Figure 6.30(c).

The Property Equivalence Matcher (PEM) and the Graph Equivalence Matcher (GEM) identified one correct relation each in all sub-datasets and none of them were identified by any of the other matchers. PEM identified *Entry* = *Publication* in sub-dataset 301303 while GEM identified *PhDThesis* = *PhdThesis*. The first relation is identified by PEM since these two classes have many data property names in common, while the second is identified by GEM since these two classes have the same parent class (*Thesis*).

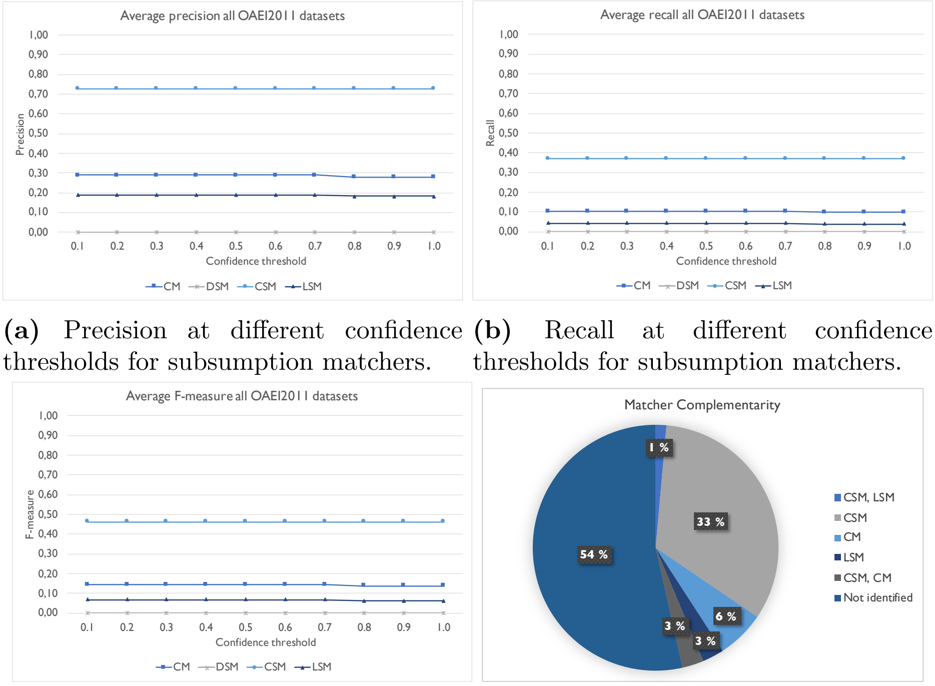
A basic string matcher based on edit distance or n-grams would also identify the *PhDThesis* = *PhdThesis* and this applies to many of the false negative relations (those relations in the reference alignment that are not identified by a matcher) in the OAEI 2011 dataset. In fact, about 55 % (6 out of the 11 relations not identified across all sub-datasets) of all false negative relations are exact string matches, but with differing capitalization. Hence, in this dataset a basic string matcher could make a strong contribution.

Evaluation of Individual Subsumption Matchers

Figure 6.31 shows average precision, recall and F-measure scores for alignments produced by the subsumption matchers for the OAEI 2011 dataset.

The Context Subsumption Matcher (CSM) obtains the highest F-measure score of the subsumption matchers in this dataset, with a score of 0.46 (at all confidence thresholds). The Compound Matcher (CM) achieves an F-measure of 0.14 and the Lexical Subsumption Matcher (LSM) achieves an F-measure of 0.07 from 0.1-0.6 that drops to 0.06 in higher thresholds. The Definition Subsumption Matcher (DSM) does not identify any correct relations in the OAEI dataset. Given the lack of natural language definitions in all sub-datasets except for 301304, that is not surprising.

In five of the sub-datasets CSM obtains the best F-measure of the individual subsumption matchers. However, in one of the sub-datasets (301303) it does not identify any correct subsumption relations. Ontology 301 has a flat structure with only one taxonomic level below the root, and as long as there is no equivalent class in ontology 301 is matched against, which there is in 301302 and 301304, CSM does not have any “anchor” from which it can infer a subsumption relation.



(a) Precision at different confidence thresholds for subsumption matchers.

(b) Recall at different confidence thresholds for subsumption matchers.

(c) F-measure at different confidence thresholds for subsumption matchers.

(d) Complementarity of the different subsumption matchers.

Figure 6.31: Average evaluation measures for individual subsumption matchers in the OAEI 2011 datasets.

A matcher complementarity map is shown in Figure 6.31(d). This map shows that more than half of the relations in all reference alignments were not identified by any subsumption matcher. Since these ontologies have several common classes, many of the false negative relations take part in more than one sub-dataset. Several of the false negative relations include class names that have a particular bibliographic reference, such as *Incollection*, *Inproceedings* and *Inbook*. In some of the ontologies they are also written as *InCollection*, *InProceedings* and *InBook*, that is, with a pascal case notation. These specialised terms are not included in WordNet, so the Lexical Subsumption Matcher is of no use here.

Furthermore, when the pascal case convention is used, the Compound Matcher splits these names into a compound modifier and a compound head, and thus infers that they are subsumed by a target concept that equals the compound head. For example, $InCollection < Collection$. Other false negative relations include two concepts which have different levels of abstraction, such as *Publication* > *Composite* or *Misc* < *Publication*.

6.4.3 Evaluation of Alignment Combination Methods

Table 6.6 shows the profile scores for each ontology pair in the OAEI 2011 dataset. Compared with the other two datasets there are some differences. There are fewer compound class names in these ontologies, hence the Compound Fraction is overall much lower than in the other datasets. Of the ontology pairs only 301-304 includes natural language definitions for both ontologies (otherwise the Definition Coverage (DC) is set to zero). Several of the ontology pairs includes few object- and data properties, so overall the Property Fraction is quite low compared to the other datasets. This results in that all profiling scores for these metrics are below the matcher selection threshold, and that the following matchers are not included in the matcher ensemble in the configuration of Profile Weight when matcher selection is enforced:

- Definition Equivalence Matcher (DEM)
- Definition Subsumption Matcher (DSM)
- Property Equivalence Matcher (PEM)
- Compound Matcher (CM)

The ontologies use a quite common terminology, using class names (or compound parts) represented in the WordNet lexicon, hence the Lexical Coverage (LC) is on average higher here than in the other to datasets.

Table 6.6: Ontology Profiling OAEI Dataset

Ontology Profiling Metric		**301-302**	**301-303**	**301-304**	**302-303**	**302-304**	**303-304**
Terminological Analysis	**Compound Fraction (CF)**	**0.25**	**0.22**	**0.14**	**0.40**	**0.32**	**0.34**
	Corpus Coverage (CC)	0.80	0.86	0.80	1.0	0.84	0.86
	Definition Coverage (DC)	**0.00**	**0.00**	**0.03**	**0.00**	**0.00**	**0.00**
Structural Analysis	**Property Fraction (PF)**	**0.19**	**0.00**	**0.33**	**0.16**	**0.48**	**0.30**
	Structural Profile (SP)	0.78	0.61	0.51	0.72	0.51	0.78
Lexical Analysis	Lexical Coverage (LC)	0.72	0.73	0.68	0.90	0.85	0.89

Figure 6.32, Figure 6.33 and Figure 6.34 show the evaluation scores for the alignment combination methods in this dataset. The two Profile Weight

variants compute the exact same alignments in this case, hence no positive contribution from the matcher selection. As the illustrations show, Profile Weight, Cut Threshold and Average Aggregation on average obtain similar precision, recall and F-measure scores until confidence threshold 0.7 when the F-measure of the Average Aggregation alignment gets reduced. Majority Vote has a higher precision than the other combination methods, but lags behind in recall and thus F-measure.

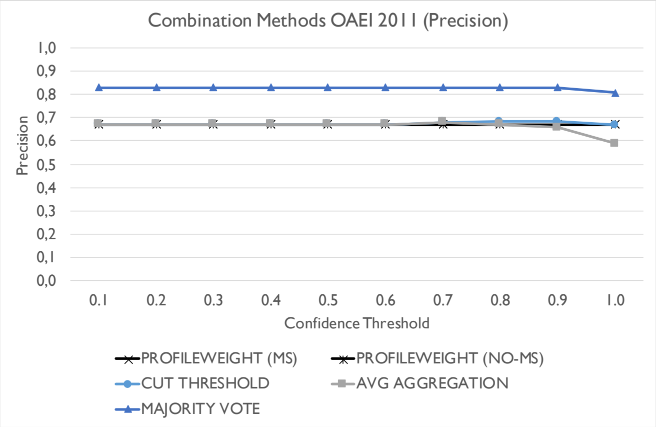


Figure 6.32: Precision of Alignment Combination Methods in the OAEI dataset.

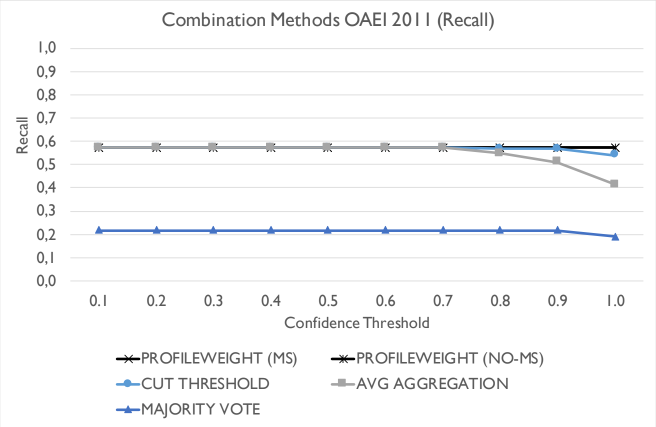


Figure 6.33: Recall of Alignment Combination Methods in the OAEI dataset.

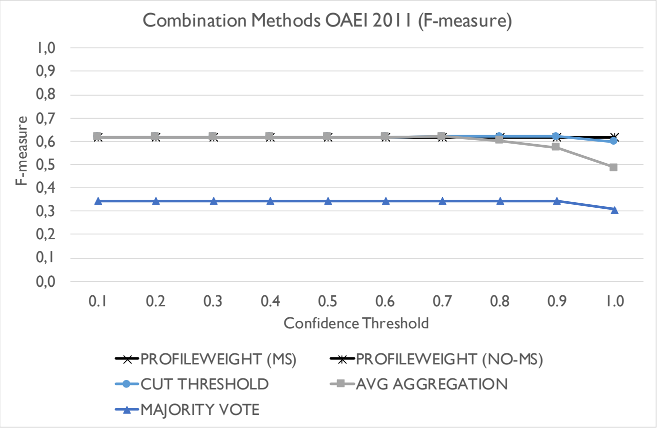


Figure 6.34: F-measure of Alignment Combination Methods in the OAEI dataset.

As illustrated in Figure 6.35, the best performing individual equivalence matcher in this dataset is the Lexical Equivalence Matcher (LEM) which obtains an F-measure of around 0.82 at confidence thresholds 0.2 - 0.5. The average F-measure when aggregating relations from all equivalence alignments using the Profile Weight method is 0.85 (from a precision of 0.89 and a recall of 0.81).

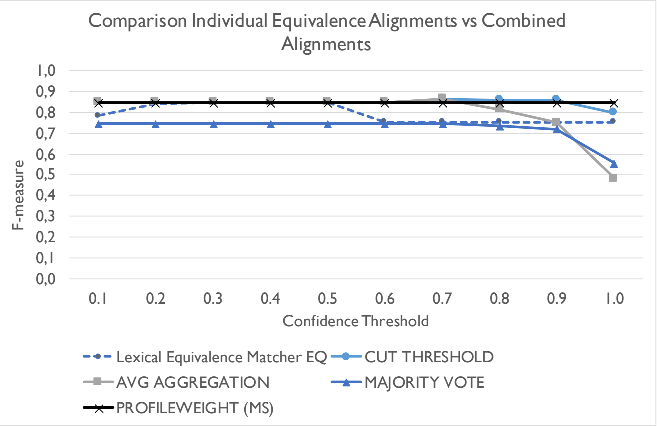


Figure 6.35: Comparison of Best Individual Equivalence Alignment and Combined Alignments in the OAEI dataset.

When using Cut Threshold to aggregate relations from the individual equivalence alignments, the best F-measure of 0.87 (precision 0.93 and recall 0.81) is obtained at confidence threshold 0.7. This is also the case for the Average Aggregation method. For the Majority Vote the highest F-measure on the final equivalence alignment is 0.75 (precision 0.91 and recall 0.63), which decreases from threshold 0.7 and onwards.

For the subsumption matching the best performing individual matcher on average is the Context Subsumption Matcher (CSM) with an F-measure of 0.46, see Figure 6.36.

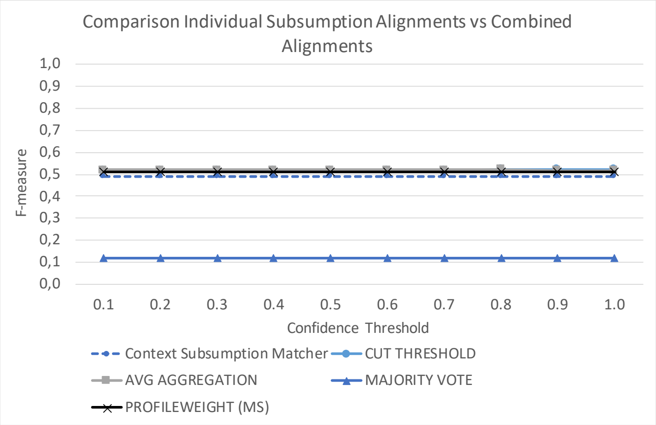


Figure 6.36: Comparison of Best Individual Subsumption Alignment and Combined Alignments in the OAEI dataset.

Here, the Profile Weight method obtains an F-measure of 0.51 (a precision of 0.56 and a recall of 0.47) when averaging the F-measure across all sub-datasets. Cut Threshold and Average Aggregation obtain an F-measure of 0.52 (0.58 in precision and 0.48 in recall) at all confidence thresholds. In this dataset, as with the other datasets, Majority Vote returns significantly lower scores in the subsumption alignment since it in most sub-datasets preserves mostly equivalence relations from the majority of the individual alignments.

So in this dataset, contrary to the other two datasets, the combination methods get a higher F-measure than the best individual alignments, both for the equivalence- and subsumption matching.

Figure 6.37, Figure 6.38 and Figure 6.39 show precision scores, recall scores and F-measure scores respectively, and illustrate how the matching systems returning both equivalence and subsumption relations compare on the OAEI dataset.

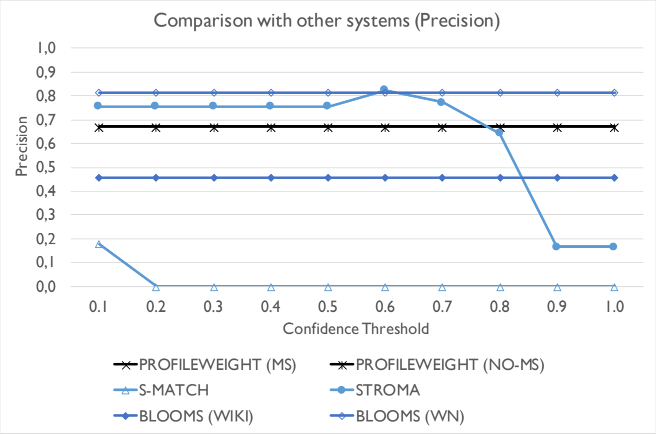


Figure 6.37: Comparing systems on precision in the OAEI dataset.

STROMA obtains the highest precision (0.83) at confidence threshold 0.6. The evaluation scores for the two Profile Weight combinations are identical, so no contribution from the matcher selection in this dataset.

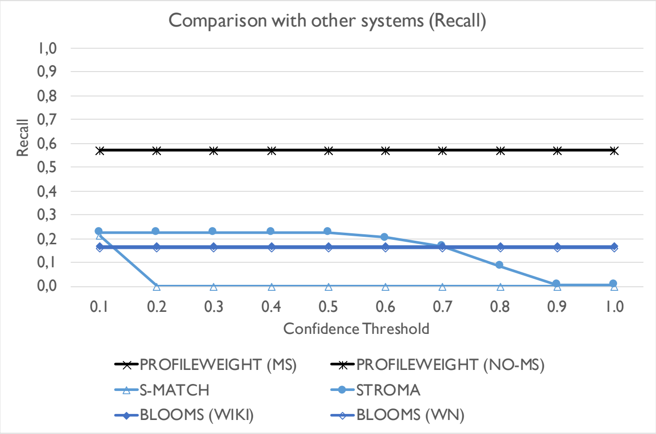


Figure 6.38: Comparing systems on recall in the OAEI dataset.

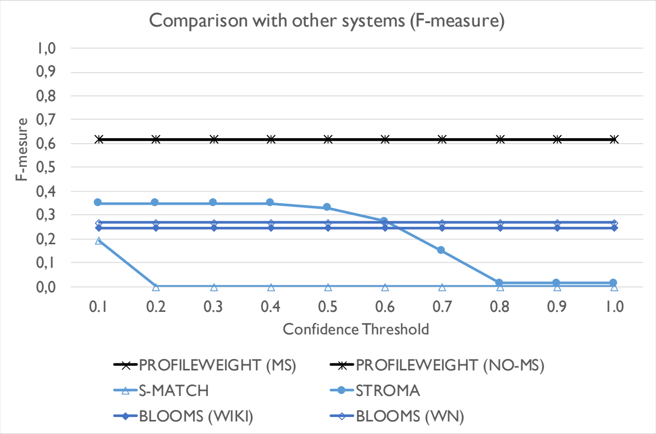


Figure 6.39: Comparing systems on F-measure in the OAEI dataset.

Figure 6.40 (semantic precision), Figure 6.41 (semantic recall) and Figure 6.42 (semantic F-measure) report the evaluation scores for the compared matching systems when using a semantic evaluation approach. In this evaluation three matching systems returning only equivalence alignments leverage the highest F-measures. AgreementMakerLight obtains an F-measure of 0.71 at confidence 0.9, LogMap achieves an F-measure of 0.68 at confidence 0.5, while the equivalence alignment configuration of Profile Weight obtains an F-measure of 0.62.

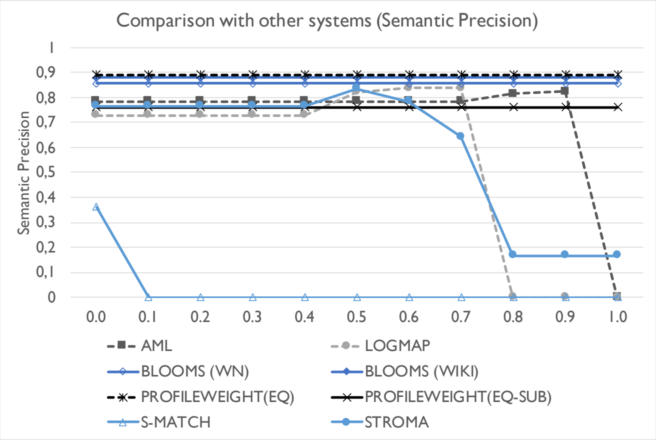


Figure 6.40: Comparing systems on semantic precision in the OAEI dataset.

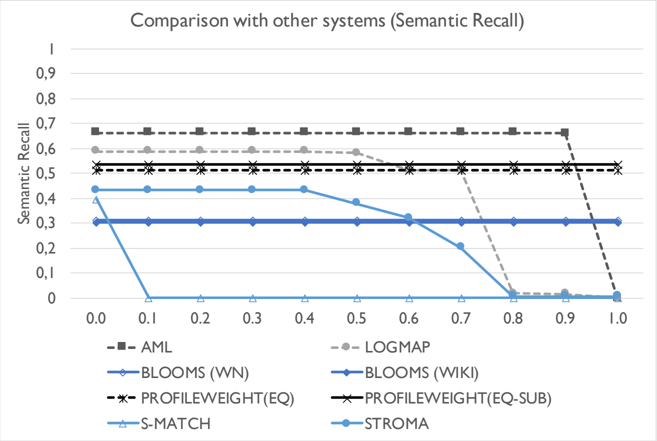


Figure 6.41: Comparing systems on semantic recall in the OAEI dataset.

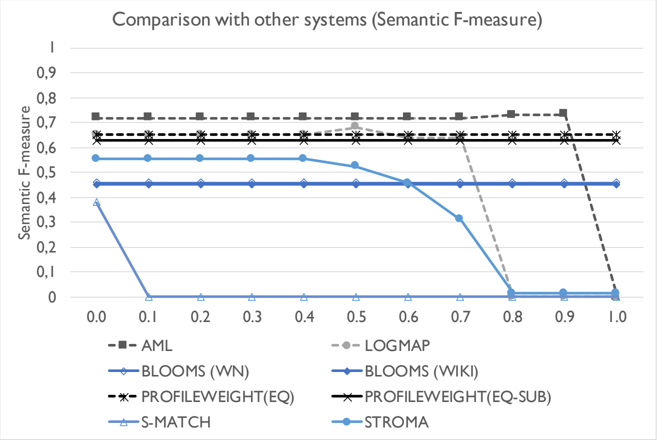


Figure 6.42: Comparing systems on semantic F-measure in the OAEI dataset.

6.4.4 Evaluation of Mismatch Detection Strategies

As with the other evaluation measures we conclude the contribution of the mismatch strategies in this dataset based on average measures. The average F-measure score of the equivalence alignment produced by the Profile Weight combination method, and across all sub-datasets, was 0.72 prior to running the mismatch detection strategies. After running the mismatch detection strategies, the F-measure score ended up at 0.85, so a significant increase in F-measure. There was no reduction in recall in any of the sub-datasets, so the mismatch detection removed only false positive relations. The largest effect was in the sub-dataset 302304, where the initial equivalence alignment obtained an F-measure of 0.59, and after the mismatch detection had removed 5 false positive relations, the F-measure ended up at 0.83. As in the other datasets it is only the Domain Mismatch Detection that makes a positive contribution.

7 Evaluation Results and Discussion

7.1 Evaluation Summary

This section summarises the key findings from the evaluation presented in the previous section.

The artefacts developed in this work has been evaluated in three diverse datasets. The first dataset, which includes two ontologies from the Air Traffic Management domain, involves technical and domain-specific terminology and fairly large ontologies. The second dataset, the Cross-Domain dataset, includes two ontologies with different focus and granularity.
Schema.org is a more general ontology used as vocabulary for web resources and Bibframe includes concepts used for describing bibliographic resources. The third dataset consists of six pairs of OAEI ontologies that all use quite generic terminology. All six ontologies are fairly small-sized and the richness in terms of properties and natural language definition varies.

7.1.1 Summary of evaluation of the Individual Matchers

The Lexical Equivalence Matcher was the equivalence matcher that performed best across all datasets, followed by the two matchers based on word embeddings, the Word Embedding Matcher and the Definition Equivalence Matcher. It is quite surprising that the Lexical Equivalence Matcher computed the best quality alignment in the ATM dataset despite the challenging terminology of this dataset. However, the word processing on the concept names contributed to overcome the limitations of WordNet with regards to limited coverage of compound words. This matcher splits compounds and uses a combination of the Jiang-Conrath semantic similarity technique and a synonym analysis in order to infer similarity between two concepts.

None of the equivalence matchers are based solely on string matching in this work, which is rather unusual. An earlier evaluation of the ATM and Cross-domain datasets suggested that using the learned word embeddings as semantic proxies from which equivalence relations could be deduced outperformed a set of string matchers tested due to an improvement in precision. The string matchers used in these experiments included ISub, n-gram, and Edit distance. However, when analysing the false negative relations in the OAEI reference alignment we see that a string matcher would probably identify about 60 % of the false negative relations since they are exact string matches, though with some variation of capitalizing of letters (e.g. *Phdthesis* - *PhDThesis* or *Inbook* - *InBook*). A lesson learned from this is that a string matcher could, based on its performance properties (i.e. fast execution), be included in a matcher ensemble, but it should probably be given less confidence than other more reliable matchers.

The Property Equivalence Matcher and the Graph Equivalence Matcher make only minor contribution in all three datasets, however the Property Equivalence Matcher identifies correct relations in the ATM and Cross-domain datasets that are not identified by any of the other matchers.

The structure-based Context Subsumption Matcher performs best of the subsumption matchers across all datasets. This matcher, which identifies subsumption relations between two concepts based on their context (parents and children), performs best in all three datasets. One can argue why there is a need for a matcher such as the Context Subsumption Matcher when the same results could be obtained by running a reasoner. The simple answer is to maintain applicability of the proposed artefacts in circumstances where using reasoning services is not feasible (e.g. schema matching scenarios). That said, as revealed by the evaluation where semantic precision and recall are used as evaluation measures, a strategy of simply using a reasoner to infer subsumption relations from equivalence relations can perform well. Of course, provided that the equivalence relations proposed in the candidate alignment are correct. The second best subsumption matcher is the Compound Matcher. Especially in the cross-domain dataset this matcher performs well. The Definition Subsumption Matcher and the Lexical Subsumption Matcher produce significantly lower F-measure scores than the other two matchers, however both capture subsumption relations that are not discovered by any other matcher.

7.1.2 Summary of evaluation of the alignment combination strategies

Profile Weight was the combination method that performed the best across all three datasets. At some confidence thresholds, the Cut Threshold and Average Aggregation methods obtained a higher F-measure in the Cross-Domain datasets, and in the OAEI 2011 dataset, these three combination methods achieved similar F-measure scores across all thresholds. The Majority Vote combination method achieved high precision in all datasets, but due to low recall it did not obtain the same level of F-measure as the other combination methods.

An important quality of the Profile Weight method is that it requires no manual configuration of a confidence threshold. It returns a combined alignment that is based on the weights assigned from the ontology profiling process. This in contrast to the Cut Threshold and Average Aggregation methods, where the quality of the combined alignment returned depends on a fixed cut threshold.

Intuitively, one would think that an ensemble of complementary matchers would obtain better quality alignments than each matcher on its own. For the equivalence matching the combination of individual alignments obtained higher F-measure scores than achieved by any of the individual matchers. But this was not the case for the subsumption matching. Here, the Context Subsumption Matcher obtained higher F-measure scores both in the ATM dataset and in the Cross-domain dataset. In the OAEI 2011 dataset, however, all combination methods except for Majority Vote, returned alignments with higher F-measure than the best subsumption matcher (Context Subsumption Matcher).

The matcher selection based on the ontology profiling had a positive effect on the quality of the combined alignments in the ATM and Cross-domain datasets. In the ATM dataset, using matcher selection for the Profile Weight gave an F-measure of 0.41 compared with an F-measure of 0.38 without matcher selection. In the Cross-domain dataset the matcher selection configuration of the Profile Weight obtained an F-measure of 0.68 compared to an F-measure of 0.65 when not selecting matchers on the basis of the profiling scores. For the OAEI dataset, the matcher selection did not have any effect. The reason for this is that the few relations identified by the matchers omitted in the matcher selection were identified by other matchers in the remaining ensemble.

7.1.3 Summary of evaluation of mismatch detection strategies

The mismatch detection strategies improved the precision in all three datasets. In the ATM dataset, running the mismatch detection strategies increased the F-measure on the equivalence alignment in the Profile Weight combination method from 0.32 to 0.52. In the Cross-Domain dataset, the strategies increased the F-measure from 0.30 to 0.65. And in the OAEI 2011 dataset, the average F-measure across all six sub-datasets increased from 0.72 to 0.85.

However, it is only the Domain Mismatch Detection strategy that contributes to this increase in F-measure, the Concept Scope Mismatch Detection is not able to identify any false positives in any of these three datasets. This contradicts the results from the experiments on these strategies reported in Section 5.5. Here, the Concept Scope Mismatch Detection filtered out 8 of the 15 false positive relations from the initial alignment produced by AgreementMakerLight when run on the ATM dataset.

The main reason for this discrepancy is the use of string matching. AgreementMakerLight uses, among other techniques, string matching techniques in its computation of equivalence relations. For example, the Word Matcher in AgreementMakerLight uses words shared by the source concept and the target concept as (partial) evidence of equivalence. This results in that for example $AircraftFlow$ and $Flow$ are considered equivalent. Furthermore, most string matchers would give a high similarity score between the two concepts $TRoute$ and $Route$. Since the implementation proposed in this thesis does not include a string matcher, but have replaced this with matchers based on word embeddings, relations such as these have not been included in the returned alignments. And this also results in that relations such as those returned by AgreementMakerLight, and which represent patterns detected by the Concept Scope Mismatch Detection technique, are not present.

7.2 Validity, Reliability and Credibility of the Research

Validity is concerned with the question of how the conclusions might be wrong, i.e. the relationship between conclusions and reality. Often, validity is distinguished in internal validity and external validity [123]. *Internal validity* relates to whether it is the treatment that actually causes the outcome, or if other external factors influence the outcome. In this research, the outcome is an alignment holding a set of correct semantic relations between concepts of two ontologies. The treatment is a semantic matching system that is capable of identifying these correct semantic relations. There are mainly two aspects to consider in this regard:

1. Whether the proposed semantic matching system, and its constituent artefacts, is actually capable of improving the quality of the produced alignment compared to if another system was used.

2. Whether using ontology profiling to automatically configure the matching system yields better quality alignments than when not using this approach.

With respect to the first aspect, the evaluation (see Chapter 6) compares the alignment quality produced by the proposed approach with the quality of alignments produced by five other semantic matching systems. Two different evaluations were performed for each of the three datasets. The first evaluation was based on “syntactic” evaluation measures, namely precision and recall as they are applied in the evaluation of information retrieval systems. In this evaluation the prototype developed in this work is compared with other semantic matching systems that compute alignments holding both equivalence- and subsumption relations. In order to also include systems that only output equivalence alignments, the second evaluation analysed evaluation scores with respect to “semantic” evaluation measures according to semantic precision and recall. Here, the proposed prototype is evaluated relative to two state-of-the-art matching systems AgreementMakerLight and LogMap in addition to the systems included in the first evaluation.

The conclusion from the first evaluation, using “syntactic” evaluation measures, is that with the proposed approach, the alignment quality is higher in terms of F-measure than all three compared systems. However, in order to produce alignments from these systems, they either had to be re-factored (BLOOMS and S-Match) or their prescribed workflow had to be changed (STROMA). From this, there is a threat to the internal validity in that such a re-factoring has resulted in changes in parameter settings or other changes that may have affected their results.

The conclusion from the second evaluation, using “semantic” evaluation measures, is that the proposed approach performs well even if the state-of-the-art system AgreementMakerLight obtains a higher F-measure in the Cross-domain dataset and the OAEI dataset.

With respect to the second aspect, we compare the quality of the alignment produced by the proposed approach with the quality of the alignment produced by the same set of matchers, but not using the results from the ontology profiling process to weigh confidence of the matchers and influence the alignment combination process. The evaluation shows that the approach using ontology profiling scores in general yields higher alignment quality in terms of F-measure than when not using this approach. In both the ATM dataset and the Cross-domain dataset the matcher selection contributes to a higher F-measure, while in the OAEI dataset the matcher selection did not have any effect. Based on these results, there is a strong indication that the proposed treatment (i.e. the matching artefacts developed in this work) has a positive effect on the semantic matching process and the outcome (the alignment quality).

External validity refers to the ability of generalising the results, that is, can the results be transferred to a wider setting outside the particular context of the study? To evaluate this, three diverse datasets were used in the experimental evaluation in order to establish an as varied and realistic context as possible. These datasets were different in terms of application domain and scope, level of generality, terminological complexity, and ontology size. The results from all datasets suggest that the proposed approach can positively contribute to alignment quality beyond a particular context. There is however a validity threat related to the ATM and the Cross-domain datasets since they, in contrast to the OAEI dataset, have not been validated by a larger community. The ATM dataset was developed in a collaboration that included experts from the ATM domain as well as researchers working in the area of semantic technologies. Following a rigorous mapping process that resulted in a manual mapping between the ATM ontologies NASA's ATMONTO and Eurocontrol's AIRM (AIRM-O), reference alignments holding subsumption- and equivalence relations were derived. Furthermore, the Cross-Domain dataset was developed by the author himself following a procedure whereby an initial equivalence alignment was generated by the AgreementMakerLight system. The author himself inspected this initial alignment, and from an analysis of the ontologies involved (Bibframe and Schema.org), the alignment was refined by adding additional equivalence relations as well as removing relations that were considered incorrect. The creation of subsumption relations for both datasets followed the same procedure as used by the Ontology Alignment Evaluation Initiative (OAEI).

With respect to the *reliability* of the research, this relates to the stability or consistency with which we measure something [123]. In the experiments de facto standard statistical measures have been used as parameters for determining the quality of an alignment produced by the individual artefacts and the prototype matching systems: precision, recall and F-measure. In order to have a valid evaluation that could put the performance of the prototype matching system in context with other relevant state-of-the-art systems semantic precision and recall were applied as evaluation metrics.

Hence, the evaluation results measured using these measures can be used in comparison with other related research.

Credibility refers to giving sufficient detail of the study to allow other researchers to replicate it [123]. In addition to describing the approach in a detailed manner, all source code and evaluation data from all experiments, including the alignments produced by these mentioned systems, are made available on-line so that the evaluation results can be traced and reproduced.

Part IV

Conclusions and Further Work

8 Conclusions and Further Work

8.1 Conclusions

The main objective of this thesis has been to: “Develop an approach for semantic matching that uses inherent characteristics of ontological models to produce an alignment that includes both equivalence and subsumption relations”.

In order to fulfil this objective, and its underlying research questions, several artefacts have been developed and evaluated in an iterative approach guided by the Design Science research paradigm. In sum, these artefacts form a prototype semantic matching system that derives inherent characteristics from the ontologies to be matched using a set of generic ontology profiling metrics that quantitatively describe terminological, structural and lexical features. Based on the derived characteristics the system selects the most relevant equivalence- and subsumption matchers to be represented in a complementary ensemble of matchers, configures the confidence threshold for the included matchers, and combines individual alignments into an optimal final alignment holding both equivalence and subsumption relations. Furthermore, a set of mismatch detection strategies have been developed to filter out false positive relations from the equivalence alignments produced and contribute to increase the precision of the final alignments returned from the system.

Evaluation results show that the proposed approach is highly competitive with state-of-the-art matching systems. Especially in one of the datasets, where the terminology used to describe the ontologies is technical and domain-specific, the approach achieves higher precision, recall and F-measure than other comparable systems.

8.2 Summary of Contributions

Building on existing knowledge from semantic matching (including schema matching and ontology matching systems research) the artefacts developed as part of this thesis include:

- Six ontology profiling metrics, which all return a quantitative score normalised between [0..1], and that are used in the weight formulation of the individual matchers.
- Five equivalence matchers, which all represent new ideas with respect to automatically identifying equivalence relations among ontology concepts.
- Four subsumption matchers, which mixes existing techniques with new ideas on how to capture subsumption relations.
- One combination method, the Profile Weight method, which combines equivalence and subsumption alignments produced by individual matchers and which considers the weight imposed from the profiling metrics.
- Two Mismatch Detection Techniques, that based on research on ontology mismatches have the capability of enhancing the precision of returned alignments without reducing the recall.
- An instantiation of all above artefacts represented as a prototype of a semantic matching system.

All source code related to these artefacts as well as all evaluation results, including the source material from which evaluation results are derived from, is made available on-line[1].

8.3 Revisiting the Research Questions

The research questions formulated in Section 1.2 are answered in the following:

[1] https://github.com/audunven

RQ1: Which ontology characteristics can guide the composition of a relevant ensemble of matchers in a semantic matching system?
During a number of iterations involving definition, testing and evaluation of numerous metrics, the ontology profiling process finally includes these metrics:

- *Compound Fraction*, which represents the fraction of how many concept names are compounds over all concept names in the two input ontologies.
- *Corpus Coverage*, which analyses how many individual tokens from the two input ontologies reside in a corpus representing word embeddings.
- *Definition Coverage*, which represents the fraction of concepts that are annotated by a natural language definition in each of the two ontologies
- *Property Fraction*, representing the fraction of classes that are associated with data- or object properties over the total number of classes in the two ontologies
- *Structural Profile*, computed as the fraction of classes that have sub- or superclasses associated with them.
- *Lexical Coverage*, which measures the ratio of class names present in the WordNet lexicon.

These profiling metrics correlate well with the equivalence- and subsumption matchers developed in this work, which is an assumption for the suggested matching process. The scores computed by the ontology profiling metrics are used for selecting matchers in an ensemble as well as determining the weights that influence the confidence values assigned to the semantic relations returned by the matchers, and consequently the combination of individual alignments.

The results from the evaluation suggests that these ontology characteristics, which encompass terminological, structural and lexical aspects, are appropriate for guiding the selection, configuration and combination of matchers.

RQ2: Which techniques can be used to automatically identify subsumption relations?
Four different subsumption matchers are implemented in this work. The

Compound Matcher, Context Subsumption Matcher and Lexical Subsumption Matcher are based on ideas from previous works (notably Arnold & Rahm [7]), while the Definition Subsumption Matcher is to the best of our knowledge new. The Definition Subsumption Matcher is based on lexico-syntactic patterns, which are commonly used to identify hyponym relations in ontology learning [62].

An analysis of the complementarity of the matchers shows that even if the Context Subsumption Matcher and the Compound Matcher make the strongest contribution in all datasets, all four techniques contribute by identifying subsumption relations that the other matchers are not able to identify.

From the evaluation of semantic precision and recall in all datasets it is clear that inferring subsumption relations from high quality equivalence alignments can contribute to identify correct subsumption relations – also relations beyond those identified by the abovementioned matchers.

RQ3: Which combination strategies are applicable when combining semantic relations - produced by an ensemble of equivalence and subsumption matchers - into a final alignment?
A large number of alignment combination strategies are proposed in the semantic matching literature. Many of them are based on the assumption that the single highest one-to-one relation should be extracted from each alignment produced by individual matchers. Clearly, such a strategy does not work for subsumption alignments, where a concept in one ontology may be related to several concepts in the other ontology.

Four different combination methods have been evaluated in this work, all applicable for combined alignments consisting of both equivalence and subsumption relations. These combination methods are:

- *Cut Threshold* [25], which on the basis of a defined cut-off threshold only allows relations above this threshold into the final alignment.
- *Average Aggregation* [25], which computes the average confidence value from all involved matchers for each relation.
- *Majority Vote* [35], which only includes those semantic relations computed by a majority of the involved matchers into the final alignment.
- *Profile Weight*, representing a weighted combination approach, using the scores from the ontology profiling process as weights for the confidence value assigned by each individual matcher.

The Profile Weight combination method performs well across all datasets and without relying on specific confidence thresholds – all relations in the final alignment are considered equally valid. The Cut Threshold and Average Aggregation combination methods sometimes outperform Profile Weight, but only at certain confidence thresholds. The Majority Vote method results in high precision, but low recall, due to the strict requirement that the majority of the involved matchers should propose a relation for it to be included in the final alignment.

RQ4: Which strategies can be used to automatically detect ontology mismatches and ultimately enhance the quality of already produced alignments?
Two mismatch detection strategies have been proposed in this work, Concept Scope Mismatch Detection and Domain Mismatch Detection. Only the Domain Mismatch Detection was able to detect mismatches in the three datasets, despite promising performance by both strategies in preliminary experiments. The Domain Mismatch Detection improved the F-measure of the input equivalence alignments significantly. For example in the Cross-Domain dataset, the F-measure went up from 0.30 to 0.65, affecting only the precision as no true positive relations were removed in the process.

8.4 Further Work

There are several avenues for further work from the research conducted in this thesis:

- Different artefacts taking part in the semantic matching process have been described and evaluated in this thesis. In order to rigorously evaluate each artefact their performance has been evaluated in isolation, not considering some of the synergies that could be obtained with a stronger integration between the artefacts throughout the matching process. For example, the discovery of a subsumption relation could also inform the discovery of novel equivalence relations and vice versa. Identifying potential synergies between the different artefacts and evaluating their effect is considered an important further work item.

- The use of word embeddings was central to the equivalence matching and replaced basic string matching techniques in this work. When inspecting the equivalence alignments produced by the two embedding matchers, we see that several of the false positives indicate some form of semantic relatedness, but other relations than equivalence. Some are subsumption relations (e.g. Carrier < Vehicle), while others have some other form of lexical-semantic relation (e.g. meronyms such as: Family *has-a* Residence). Some work has been done on automatic detection of hyponyms in natural text using word embeddings, e.g. [160, 40, 115] and investigating how embeddings could infer subsumption relations and other lexical semantic relations would be a natural extension to the work proposed.

- One line of work in semantic matching has been automated identification of background knowledge [38]. One interesting idea would be to automatically identify appropriate natural text corpora that could be used to derive word embeddings.

- In this work WordNet was used as an external lexical resource. Using other lexical resources, such as the semantic network BabelNet[2] and the related BabelFy[3] which performs multilingual word sense disambiguation and entity linking, should be investigated. BabelNet includes semantic description of concepts from various sources, including WordNet. Some small experiments were conducted as part of this work, but the conclusion from these experiments is that using some of the other lexical resources included in BabelNet (e.g. Wikidata) introduces additional false positives and longer run-time. However, BabelNet appears to be a more active development than WordNet, hence increased functionality, possibly useful for semantic matching, is expected. Furthermore, and as briefly introduced in the Related Work in Section 3.1, WebIsA is a knowledge base of hyponym relations extracted from Common Crawl[4], a freely available corpus crawled from the Web. Investigating to what extent this knowledge based can be used in subsumption matching would be an interesting future work item.

- One of the subsumption matchers, the Definition Subsumption Matcher, used lexico-syntactic patterns in the natural language definition of the ontology concepts to infer subsumption relations between them. Five different patterns were applied, and these patterns helped to detect subsumption relations that were not discovered by any other matcher. However, an analysis of other relevant pattern could be performed,

[2] https://babelnet.org/
[3] http://babelfy.org/
[4] http://commoncrawl.org/

and techniques for enriching concepts that are sparsely documented in definitions would be a feasible way forward.

- A more sophisticated way of de-compounding could possibly enhance the results of the Compound Matcher as well as other matchers using the properties of compounds as part of the matching process. De-compounding is challenging for several reasons (see Section 5.3.1) and techniques with more accuracy and reliability should be developed.

- Within this work, run-time performance has not been an issue, and most, if not all, matchers could most likely be made more efficient in that regard. This is something that should be addressed by future extensions of the individual matchers and the proposed matching system prototype. Especially when even larger ontologies are to be matched, the run-time performance of the current matchers could represent a bottleneck. *Large-scale ontology matching* is an active sub-research area to ontology matching, and despite the fact that many techniques for making the matching process more efficient has been proposed, this is still considered a general issue in matching [35].

Part V

Appendices

A Publications

This thesis is supported by 8 publications that are presented in the following. Of these 3 are CORE-B papers, 2 are CORE-C papers, 1 is a PhD symposium paper in a CORE-A conference, 1 is a workshop paper (The Ontology Matching Workshop) and 1 paper was in an applied conference. See http://portal.core.edu.au/conf-ranks/.

- **Vennesland, A., e-Document Standards as Background Knowledge in Context-Based Ontology Matching. In European Semantic Web Conference (pp. 806-816). Springer, Cham, 2015.** The work described in this paper investigated how external background knowledge can be used to detect correct equivalence relations between ontology concepts. The source of background knowledge was the e-Document standard Universal Business Language (UBL) from the transport logistics domain. This experiences from this work triggered the exploration of using word embeddings as a more sophisticated background knowledge source for automated detection of semantic relations between ontology concepts.

- **Vennesland, A., Matcher composition for identification of subsumption relations in ontology matching. In Proceedings of the International Conference on Web Intelligence (pp. 154-161), ACM, 2017.** This paper describes how ontology profiling can contribute to automate matcher selection and combination when aligning ontologies. The focus in this paper is on subsumption matching.

- **Vennesland, A., and Aalberg, T., False-Positive Reduction

in Ontology Matching Based on Concepts' Domain Similarity. In International Conference on Theory and Practice of Digital Libraries (pp. 344-348), Springer, Cham, 2018. This paper describes how the WordNet Domains classification can be applied to detect domain dissimilarity between ontology concepts and thereby rule out false positive semantic relations in already produced alignments. The dataset used in the work described by this paper is the Cross-domain dataset (equivalence relations only) used in the thesis.

- **Vennesland, A., Gorman, J., Wilson, S., Neumayr, B., Schuetz, C.G., Automated Compliance Verification in ATM using Principles from Ontology Matching. In Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KEOD) (pp. 39-50), 2018. This paper received the "“Best Paper in Conference Award" at KEOD 2018.** This paper emerged from the BEST EU project, where one objective was to develop a prototype software tool for assisting human users in checking compliance between exchange models in ATM and the ATM Information Reference Model (AIRM) which is the standard information model for information exchange in the ATM domain. The paper also describes the AIRM-O ontology, one of the ontologies used in the ATM dataset, which was developed in this research project. Furthermore, this work investigated how the results from the ontology profiling could impact on selecting the most appropriate matchers for a given ontology matching task, and some of the techniques used by the matchers in this thesis originate from this work.

- **Gringinger, E., Keller, R.M., Vennesland, A., Schuetz, C.G., Neumayr, B. A Comparative Study of Two Complex Ontologies in Air Traffic Management. The 38th AIAA/IEEE Digital Avionics Systems Conference (DASC), 2019.** This paper describes the analysis performed when identifying semantic relations between the AIRM-O and ATMONTO ontologies (the ATM dataset in this thesis). The focus of the paper is on the human effort of identifying the semantic relations as well as categorising the ontology mismatches that exist between the two ATM ontologies.

- **Vennesland, A., Keller, R.M., Schuetz, C.G., Gringinger, E., Neumayr, B. Matching Ontologies for Air Traffic Management: A Comparison and Reference Alignment of the AIRM**

and NASA ATM Ontologies. In proceedings of the 14th International Workshop on Ontology Matching co-located with the 18th International Semantic Web Conference (ISWC 2019) Auckland, New Zealand (pp. 1-12), 2019. This paper focuses on the development of the ATM dataset used in this thesis. The paper describes the development of the reference alignment, which holds both equivalence and subsumption relations, between the AIRM-O and ATMONTO ontologies.

- **Decourselle, J., Vennesland, A., Aalberg, T., Duchateau, F., and Lumineau, N. A novel vision for navigation and enrichment in cultural heritage collections. In East European Conference on Advances in Databases and Information Systems (pp. 488-497), Springer, Cham, 2015.** This paper describes an approach for developing thematic knowledge bases (TKB) for cultural heritage information. The approach is based on gathering information about a particular topic (e.g. an actor) from different sources (both linked open data sources and natural text repositories). Semantic matching (ontology- and entity matching) is used to identify additional sources of information to enrich the existing TKB and to de-duplicate the knowledge residing in the TKB.

- **Vennesland, A., de Man, J.C., Halland Haro, P., Arica, E., Oliveira, M., Towards a semantic matchmaking algorithm for capacity exchange in manufacturing supply chains, In Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KEOD), 2019.** The MANU-SQUARE project aims to deploy a marketplace for the matchmaking of offer and demand of manufacturing resources. A fundamental component in this marketplace is a semantic matchmaking algorithm that based on an ontology describing manufacturing resources is capable of finding the best supplier given an explicit resource demand. This paper reports on an evaluation of four different semantic similarity techniques to use for this algorithm.

Bibliography

[1] Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. ACM sigmod record 22(2), 207–216 (1993)

[2] Agrawal, R., Srikant, R., et al.: Fast algorithms for mining association rules. In: Proc. 20th int. conf. very large data bases, VLDB. vol. 1215, pp. 487–499 (1994)

[3] Alpaydin, E.: Introduction to Machine Learning. The MIT Press, 2nd edn. (2010)

[4] Ameri, F., Patil, L.: Digital manufacturing market: a semantic web-based framework for agile supply chain deployment. Journal of Intelligent Manufacturing 23(5), 1817–1832 (2012)

[5] Anam, S., Kim, Y.S., Kang, B.H., Liu, Q.: Review of ontology matching approaches and challenges. International journal of Computer Science and Network Solutions 3(3), 1–27 (2015)

[6] Angermann, H., Ramzan, N.: Taxonomy Matching Using Background Knowledge. Springer (2017)

[7] Arnold, P., Rahm, E.: Enriching ontology mappings with semantic relations. Data & Knowledge Engineering 93, 1–18 (2014)

[8] Artale, A., Franconi, E., Guarino, N., Pazzi, L.: Part-whole relations in object-centered systems: An overview. Data & Knowledge Engineering 20(3), 347–383 (1996)

[9] Bellahsene, Z., Bonifati, A., Rahm, E.: Schema Matching and Mapping. Springer-Verlag Berlin Heidelberg (2011)

[10] Bishop, C.M.: Pattern recognition and machine learning. Springer Science+ Business Media (2006)

[11] Brank, J., Grobelnik, M., Mladenic, D.: A survey of ontology evaluation techniques. In: Proceedings of the conference on data mining and data warehouses (SiKDD 2005). pp. 166–170. Citeseer Ljubljana, Slovenia (2005)

[12] Budanitsky, A., Hirst, G.: Evaluating WordNet-based measures of lexical semantic relatedness. Computational Linguistics 32(1), 13–47 (2006)

[13] Chalupsky, H.: Ontomorph: A translation system for symbolic knowledge. In: KR. pp. 471–482 (2000)

[14] Cheatham, M., Hitzler, P.: The role of string similarity metrics in ontology alignment. Tech. rep., Wright State University CORE Scholar (2013)

[15] Cheatham, M., Hitzler, P.: String similarity metrics for ontology alignment. In: International Semantic Web Conference. pp. 294–309. Springer (2013)

[16] Cheatham, M., Hitzler, P.: StringsAuto and MapSSS results for OAEI 2013. In: Proceedings of the 8th International Workshop on Ontology Matching co-located with the 12th International Semantic Web Conference (ISWC 2013) (2013)

[17] Cheatham, M., Hitzler, P.: Conference v2. 0: An uncertain version of the OAEI conference benchmark. In: International Semantic Web Conference. pp. 33–48. Springer (2014)

[18] Cheatham, M., Hitzler, P.: The properties of property alignment. In: Proceedings of the 9th International Conference on Ontology Matching-Volume 1317. pp. 13–24. CEUR-WS. org (2014)

[19] Cross, V.: Semantic similarity: A key to ontology alignment. In: Ontology Matching: OM-2018: Proceedings of the ISWC Workshop. p. 61 (2018)

[20] Cruz, I.F., Antonelli, F.P., Stroe, C.: Efficient selection of mappings and automatic quality-driven combination of matching methods. In: Proceedings of the 4th International Conference on Ontology Matching-Volume 551. pp. 49–60. Citeseer (2009)

[21] Cruz, I.F., Fabiani, A., Caimi, F., Stroe, C., Palmonari, M.: Automatic configuration selection using ontology matching task profiling. In: Extended Semantic Web Conference. pp. 179–194. Springer (2012)

[22] Cruz, I.F., Palmonari, M., Caimi, F., Stroe, C.: Building linked ontologies with high precision using subclass mapping discovery. Artificial Intelligence Review 40(2), 127–145 (Aug 2013), http://link.springer.com/10.1007/s10462-012-9363-x

[23] David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The alignment API 4.0. Semantic web 2(1), 3–10 (2011)

[24] Djeddi, W.E., Khadir, M.T.: A dynamic multistrategy ontology alignment framework based on semantic relationships using WordNet. In: In Proc of the 3rd International Conference on Computer Science and its Applications (CIIA 11), 13- 15 December, Saida. Citeseer (2011)

[25] Do, H.H., Rahm, E.: COMA: a system for flexible combination of schema matching approaches. In: Proceedings of the 28th international conference on Very Large Data Bases. pp. 610–621. VLDB Endowment (2002)

[26] Duchateau, F., Coletta, R., Bellahsene, Z., Miller, R.J.: (not) yet another matcher. In: Proceedings of the 18th ACM conference on Information and knowledge management. pp. 1537–1540. ACM (2009)

[27] Eckert, K., Meilicke, C., Stuckenschmidt, H.: Improving ontology matching using meta-level learning. In: European Semantic Web Conference. pp. 158–172. Springer (2009)

[28] Ehrig, M., Sure, Y.: Ontology mapping–an integrated approach. In: European semantic web symposium. pp. 76–91. Springer (2004)

[29] Eurocontrol: Aeronautical Information Exchange Model 5.1. http://www.aixm.aero/ (2014), [Online; accessed 08-December-2018]

[30] Eurocontrol: Flight Information Exchange Model 4.0. http://www.fixm.aero (2016), [Online; accessed 08-December-2018]

[31] EUROCONTROL: ATM Information Reference Model v4.1.0 (2017), http://airm.aero/

[32] Eurocontrol: Weather Information Exchange Model 2.0. http://www.wxxm.aero (2017), [Online; accessed 08-December-2018]

[33] Euzenat, J.: Semantic precision and recall for ontology alignment evaluation. In: IJCAI. vol. 7, pp. 348–353 (2007)

[34] Euzenat, J.: Algebras of Ontology Alignment Relations. In: International Semantic Web Conference. Springer Berlin Heidelberg (2008)

[35] Euzenat, J., Shvaiko, P.: Ontology matching. en. 2nd. Heidelberg (DE): Springer-Verlag p. 8 (2013)

[36] Faria, D., Pesquita, C., Santos, E., Cruz, I.F., Couto, F.M.: AgreementMakerLight results for OAEI 2013. In: OM. pp. 101–108 (2013)

[37] Faria, D., Pesquita, C., Santos, E., Cruz, I.F., Couto, F.M.: AgreementMakerLight 2.0: Towards Efficient Large-Scale Ontology Matching. In: International Semantic Web Conference (Posters & Demos). pp. 457–460 (2014)

[38] Faria, D., Pesquita, C., Santos, E., Cruz, I.F., Couto, F.M.: Automatic background knowledge selection for matching biomedical ontologies. PloS one 9(11), e111226 (2014)

[39] Faria, D., Pesquita, C., Santos, E., Palmonari, M., Cruz, I.F., Couto, F.M.: The AgreementMakerLight ontology matching system. In: OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". pp. 527–541. Springer (2013)

[40] Fu, R., Guo, J., Qin, B., Che, W., Wang, H., Liu, T.: Learning semantic hierarchies via word embeddings. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 1199–1209 (2014)

[41] Gal, A., Shvaiko, P.: Advances in ontology matching. In: Advances in web semantics i, pp. 176–198. Springer (2008)

[42] Gangemi, A., Catenacci, C., Ciaramita, M., Lehmann, J.: Ontology evaluation and validation. Proceedings of the 3rd European Semantic Web Conference (ESWC2006) 3, 140–154 (2006)

[43] Gangemi, A., Catenacci, C., Ciaramita, M., Lehmann, J.: A theoretical framework for ontology evaluation and validation. In: SWAP. vol. 166, p. 16. Citeseer (2005)

[44] García, J., García-Peñalvo, F.J., Therón, R.: A survey on ontology metrics. Communications in Computer and Information Science 111 CCIS(PART 1), 22–27 (2010)

[45] Gella, S., Strapparava, C., Nastase, V.: Mapping WordNet Domains, WordNet Topics and Wikipedia Categories to generate multilingual domain specific resources. In: LREC. pp. 1117–1121 (2014)

[46] Giunchiglia, F., Autayeu, A., Pane, J.: S-Match: an open source framework for matching lightweight ontologies. Semantic Web 3(3), 307–317 (2012)

[47] Giunchiglia, F., Shvaiko, P., Yatskevich, M.: S-match: an algorithm and an implementation of semantic matching. In: European semantic web symposium. pp. 61–75. Springer (2004)

[48] Giunchiglia, F., Shvaiko, P., Yatskevich, M.: Discovering missing background knowledge in ontology matching. In: ECAI. vol. 141, pp. 382–386 (2006)

[49] Gracia, J., Bernad, J., Mena, E.: Ontology matching with CIDER: Evaluation report for OAEI 2011. Ontology Matching p. 126 (2011)

[50] Gras, R., Kuntz, P.: An overview of the Statistical Implicative Analysis (SIA) development. In: Statistical implicative analysis, pp. 11–40. Springer (2008)

[51] Grau, B.C., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. Journal of Artificial Intelligence Research 31, 273–318 (2008)

[52] Green, R., Bean, C.A., Myaeng, S.H.: The semantics of relationships: an interdisciplinary perspective, vol. 3. Springer Science & Business Media (2013)

[53] Gringinger, E., Keller, R.M., Vennesland, A., Schuetz, C., Neumayr, B.: A Comparative Study of Two Complex Ontologies in Air Traffic Management. In: Proceedings from the 38th AIAA/IEEE Digital Avionics Systems Conference (DASC). IEEE (2019)

[54] Guarino, N.: Formal ontology in information systems: Proceedings of the first international conference (FOIS'98), June 6-8, Trento, Italy, vol. 46. IOS press (1998)

[55] Guarino, N., Welty, C.: Ontological analysis of taxonomic relationships. In: International Conference on Conceptual Modeling. pp. 210–224. Springer (2000)

[56] Gulić, M., Vrdoljak, B., Banek, M.: Cromatcher: An ontology matching system based on automated weighted aggregation and iterative final alignment. Web Semantics: Science, Services and Agents on the World Wide Web 41, 50–71 (2016)

[57] Halevy, A.: Why your data won't mix. Queue 3(8), 50–58 (2005)

[58] Hamdi, F., Reynaud, C., Safar, B.: Pattern-based mapping refinement. In: International Conference on Knowledge Engineering and Knowledge Management. pp. 1–15. Springer (2010)

[59] Hamdi, F., Safar, B., Reynaud, C., Zargayouna, H.: Alignment-based partitioning of large-scale ontologies. In: Advances in knowledge discovery and management, pp. 251–269. Springer (2010)

[60] Hamdi, F., Zargayouna, H., Safar, B., Reynaud, C.: TaxoMap in the OAEI 2008 alignment contest. In: The Third International Workshop on Ontology Matching. pp. 206–213 (2008)

[61] Hamming, R.W.: Error detecting and error correcting codes. The Bell system technical journal 29(2), 147–160 (1950)

[62] Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th conference on Computational linguistics-Volume 2. pp. 539–545. Association for Computational Linguistics (1992)

[63] Hevner, A.R.: A three cycle view of design science research. Scandinavian journal of information systems 19(2), 4 (2007)

[64] Hevner, A.R., March, S.T.: The information systems research cycle. Computer 36(11), 111–113 (2003)

[65] Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Quarterly pp. 75–105 (2004)

[66] Horridge, M., Bechhofer, S.: The OWL API: A Java API for OWL ontologies. Semantic Web 2(1), 11–21 (2011)

[67] Hu, W., Jian, N., Qu, Y., Wang, Y.: Gmo: A graph matching for ontologies. In: Proceedings of K-CAP Workshop on Integrating Ontologies. pp. 41–48 (2005)

[68] Hu, W., Qu, Y.: Falcon-AO: A practical ontology matching system. Journal of web semantics 6(3), 237–239 (2008)

[69] Hu, W., Qu, Y., Cheng, G.: Matching large ontologies: A divide-and-conquer approach. Data & Knowledge Engineering 67(1), 140–160 (Oct 2008), http://linkinghub.elsevier.com/retrieve/pii/S0169023X08000864

[70] Jaccard, P.: Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. Bull Soc Vaudoise Sci Nat 37, 241–272 (1901)

[71] Jain, P., Hitzler, P., Sheth, A.P., Verma, K., Yeh, P.Z.: Ontology alignment for Linked Open Data. In: International semantic web conference. pp. 402–417. Springer (2010)

[72] Jaro, M.A.: Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. Journal of the American Statistical Association 84(406), 414–420 (1989)

[73] Järvenpää, E., Siltala, N., Hylli, O., Lanz, M.: Product model ontology and its use in capability-based matchmaking. Procedia CIRP 72(1), 1094–1099 (2018)

[74] Jérôme, D., Guillet, F., Briand, H.: Association rule ontology matching approach. International Journal on Semantic Web and Information Systems 3(2), 27 (2007)

[75] Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy. arXiv preprint cmp-lg/9709008 (1997)

[76] Jiménez-Ruiz, E., Agibetov, A., Chen, J., Samwald, M., Cross, V.: Dividing the ontology alignment task with semantic embeddings and logic-based modules. arXiv preprint arXiv:2003.05370 (2020)

[77] Jiménez-Ruiz, E., Agibetov, A., Samwald, M., Cross, V.: We divide, you conquer: From large-scale ontology alignment to manageable subtasks. In: Ontology Matching: OM-2018: Proceedings of the ISWC Workshop. p. 13 (2018)

[78] Jiménez-Ruiz, E., Grau, B.C.: Logmap: Logic-based and scalable ontology matching. In: International Semantic Web Conference. pp. 273–288. Springer (2011)

[79] Kamel, M., Schmidt, D., Trojahn, C., Vieira, R.: Exploiting BabelNet for generating subsumption. In: 13th International Workshop on Ontology Matching co-located with the 17th ISWC (OM 2018). vol. 2288, pp. 216–217 (2018)

[80] Keller, R.M.: NASA Air Traffic Management Ontology (ATMONTO) (Mar 2018), https://data.nasa.gov/ontologies/atmonto/

[81] Keller, R.M., Ranjan, S., Wei, M.Y., Eshow, M.M.: Semantic Representation and Scale-up of Integrated Air Traffic Management Data. In: Proceedings of the International Workshop on Semantic Big Data. pp. 4:1–4:6. SBD '16, ACM, New York, NY, USA (2016), http://doi.acm.org/10.1145/2928294.2928296

[82] Khoo, C.S., Na, J.C.: Semantic relations in information science. Annual review of information science and technology 40(1), 157–228 (2006)

[83] Klein, M.: Combining and relating ontologies: an analysis of problems and solutions. In: IJCAI-2001 Workshop on ontologies and information sharing. pp. 53–62. USA. (2001)

[84] Krovetz, R.: Homonymy and polysemy in information retrieval. In: Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics. pp. 72–79. Association for Computational Linguistics (1997)

[85] Lantow, B.: OntoMetrics: Putting Metrics into Use for Ontology Evaluation. In: KEOD. pp. 186–191 (2016)

[86] Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet physics doklady 10(8), 707–710 (1966)

[87] Li, J., Tang, J., Li, Y., Luo, Q.: RiMOM: A dynamic multistrategy ontology alignment framework. IEEE Transactions on Knowledge and Data Engineering 21(8), 1218–1232 (2009)

[88] Lin, D., et al.: An information-theoretic definition of similarity. In: Icml. vol. 98, pp. 296–304. Citeseer (1998)

[89] Lin, F., Sandkuhl, K.: A survey of exploiting WordNet in ontology matching. In: IFIP International Conference on Artificial Intelligence in Theory and Practice. pp. 341–350. Springer (2008)

[90] Liu, B.: Web data mining: exploring hyperlinks, contents, and usage data. Springer Science & Business Media (2007)

[91] Mao, M., Peng, Y., Spring, M.: A Harmony based adaptive ontology mapping approach. In: SWWS. pp. 336–342 (2008)

[92] Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A., Schneider, L.: Wonderweb deliverable D17: the Wonderweb library of foundational ontologies, technical report. Tech. rep., The Wonderweb Project (2002)

[93] Massmann, S., Raunich, S., Aumüller, D., Arnold, P., Rahm, E.: Evolution of the COMA match system. In: Proceedings of the 6th International Conference on Ontology Matching-Volume 814. pp. 49–60. CEUR-WS. org (2011)

[94] Meilicke, C., Stuckenschmidt, H.: Analyzing mapping extraction approaches. In: Proceedings of the 2nd International Conference on Ontology Matching-Volume 304. pp. 25–36. CEUR-WS. org (2007)

[95] Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In: Proceedings 18th International Conference on Data Engineering. pp. 117–128. IEEE (2002)

[96] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. CoRR abs/1301.3 (2013)

[97] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed Representations of Words and Phrases and their Compositionality. In: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States. pp. 3111–3119 (2013)

[98] Miller, G., Fellbaum, C.: Wordnet: An electronic lexical database (1998)

[99] Mochol, M., Jentzsch, A.: Towards a rule-based matcher selection. In: International Conference on Knowledge Engineering and Knowledge Management. pp. 109–119. Springer (2008)

[100] Mochol, M., Jentzsch, A., Euzenat, J.: Applying an analytic method for matching approach selection. In: Proceedings of the 1st International Conference on Ontology Matching-Volume 225. pp. 37–48. CEUR-WS. org (2006)

[101] Nezhadi, A.H., Shadgar, B., Osareh, A.: Ontology alignment using machine learning techniques. International Journal of Computer Science & Information Technology 3(2), 139 (2011)

[102] Ngo, D., Bellahsene, Z.: Overview of YAM++—(not) Yet Another Matcher for ontology alignment task. Web Semantics: Science, Services and Agents on the World Wide Web 41, 30–49 (2016)

[103] Ngo, D., Bellahsene, Z., Coletta, R.: YAM++-a combination of graph matching and machine learning approach to ontology alignment task. Journal of Web Semantics 16, 16 (2012)

[104] Ngo, D., Bellahsene, Z., Coletta, R.: A generic approach for combining linguistic and context profile metrics in ontology matching. In: OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". pp. 800–807. Springer (2011)

[105] Ngo, D., Bellahsene, Z., Todorov, K.: Opening the black box of ontology matching. In: Extended Semantic Web Conference. pp. 16–30. Springer (2013)

[106] Noy, N.F., Hafner, C.D.: The state of the art in ontology design: A survey and comparative review. AI magazine 18(3), 53 (1997)

[107] Oates, B.J.: Researching information systems and computing. Sage (2005)

[108] OMG: Ontology Definition Metamodel v1.1 (2014), https://www.omg.org/spec/ODM/1.1/

[109] Otero-Cerdeira, L., Rodríguez-Martínez, F.J., Gómez-Rodríguez, A.: Ontology matching: A literature review. Expert Systems with Applications 42(2), 949–971 (2015)

[110] Pedersen, T.: Information content measures of semantic similarity perform better without sense-tagged text. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. pp. 329–332 (2010)

[111] Petrakis, E.G., Varelas, G., Hliaoutakis, A., Raftopoulou, P.: Design and evaluation of semantic similarity measures for concepts stemming from the same or different ontologies. In: 4th Workshop on Multimedia Semantics (WMS'06). pp. 44–52 (2006)

[112] Peukert, E., Massmann, S., Koenig, K.: Comparing similarity combination methods for schema matching. Gi jahrestagung (1) 10, 692–701 (2010)

[113] Pirró, G., Talia, D.: UFOme: An ontology mapping system with strategy prediction capabilities. Data & Knowledge Engineering 69(5), 444–471 (2010)

[114] Po, L., Bergamaschi, S.: Automatic lexical annotation applied to the SCARLET ontology matcher. In: Asian Conference on Intelligent Information and Database Systems. pp. 144–153. Springer (2010)

[115] Pocostales, J.: Nuig-unlp at semeval-2016 task 13: A simple word embedding-based approach for taxonomy extraction. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016). pp. 1298–1302 (2016)

[116] Porn, A., Huve, C., Peres, L., Direne, A.: A systematic literature review of OWL ontology evaluation. Proceedings of the 15th International Conference WWW/Internet 2016 pp. 67–74 (2016)

[117] Portisch, J., Paulheim, H.: ALOD2Vec matcher. In: Ontology Matching: OM-2018: Proceedings of the ISWC Workshop. p. 132 (2018)

[118] Poveda-Villalón, M., Suárez-Figueroa, M.C., Gómez-Pérez, A.: Validating ontologies with OOPS! Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 7603 LNAI, 267–281 (2012)

[119] Prins, H.: Matching ontologies with distributed word embeddings. Radboud Universiteit (2016)

[120] Rahm, E.: Towards large-scale schema and ontology matching. In: Schema matching and mapping, pp. 3–27. Springer (2011)

[121] Resnik, P.: Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. Journal of artificial intelligence research 11, 95–130 (1999)

[122] Ristoski, P., Paulheim, H.: Rdf2vec: Rdf graph embeddings for data mining. In: International Semantic Web Conference. pp. 498–514. Springer (2016)

[123] Robson, C.: Real world research, vol. 3. Wiley Chichester (2011)

[124] Rodríguez, M.A., Egenhofer, M.J.: Determining semantic similarity among entity classes from different ontologies. IEEE transactions on knowledge and data engineering 15(2), 442–456 (2003)

[125] Sabou, M., d'Aquin, M., Motta, E.: Exploring the semantic web as background knowledge for ontology matching. In: Journal on data semantics XI, pp. 156–190. Springer (2008)

[126] Salton, G., McGill, M.J.: Introduction to modern information retrieval. mcgraw-hill (1983)

[127] Schmid, H.: Improvements in part-of-speech tagging with an application to German. In: Natural language processing using very large corpora, pp. 13–25. Springer (1999)

[128] Schönböck, J., Altmann, J., Kapsammer, E., Kimmerstorfer, E., Pröll, B., Retschitzegger, W., Schwinger, W.: A semantic matchmaking framework for volunteering marketplaces. In: World Conference on Information Systems and Technologies. pp. 701–711. Springer (2018)

[129] Schopman, B.A., Wang, S., Schlobach, S.: Deriving concept mappings through instance mappings. In: Asian Semantic Web Conference. pp. 122–136. Springer (2008)

[130] Seco, N., Veale, T., Hayes, J.: An intrinsic information content metric for semantic similarity in WordNet. In: Ecai. vol. 16, p. 1089 (2004)

[131] Shvaiko, P., Euzenat, J.: Ten challenges for ontology matching. In: OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". pp. 1164–1182. Springer (2008)

[132] Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Transactions on knowledge and data engineering 25(1), 158–176 (2013)

[133] Shvaiko, P., Giunchiglia, F., Yatskevich, M.: Semantic matching with S-Match. In: Semantic Web Information Management, pp. 183–202. Springer (2010)

[134] Simon, H.A.: The Sciences of the Artificial. MIT Press, 3 edn. (1996)

[135] Smith, B., Almeida, M., Bona, J.: Basic formal ontology 2.0: Specification and user's guide (2015), https://github.com/BFO-ontology/BFO

[136] Spiliopoulos, V., Valarakos, A.G., Vouros, G.A., Karkaletsis, V.: Learning subsumption relations with CS: a classification based method for the alignment of ontologies. In: Proceedings of the 2nd International Conference on Ontology Matching-Volume 304. pp. 316–320. CEUR-WS. org (2007)

[137] Spiliopoulos, V., Vouros, G.A., Karkaletsis, V.: On the discovery of subsumption relations for the alignment of ontologies. Journal of Web Semantics 8(1), 69–88 (2010)

[138] Staab, S., Studer, R.: Handbook on Ontologies. Springer Science & Business Media (2013)

[139] Stoilos, G., Stamou, G., Kollias, S.: A string metric for ontology alignment. In: International Semantic Web Conference. pp. 624–637. Springer (2005)

[140] Storey, V.C.: Understanding semantic relationships. The VLDB Journal 2(4), 455–488 (1993)

[141] Su, X.: Semantic Enrichment for Ontology Mapping. Norwegian University of Science and Technology (2004)

[142] Su, X., Gulla, J.A.: Semantic enrichment for ontology mapping. In: International Conference on Application of Natural Language to Information Systems. pp. 217–228. Springer (2004)

[143] Suchanek, F.M., Abiteboul, S., Senellart, P.: Paris: Probabilistic alignment of relations, instances, and schema. Proceedings of the VLDB Endowment 5(3), 157–168 (2011)

[144] Tan, H., Lambrix, P.: A method for recommending ontology alignment strategies. In: The Semantic Web, pp. 494–507. Springer (2007)

[145] Tang, J., Li, J., Liang, B., Huang, X., Li, Y., Wang, K.: Using bayesian decision for ontology mapping. Web Semantics: Science, Services and Agents on the World Wide Web 4(4), 243–262 (2006)

[146] Tartir, S., Arpinar, I.B., Moore, M., Sheth, A.P., Aleman-Meza, B.: OntoQA: Metric-based ontology quality analysis. In: IEEE ICDM 2005 Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources (2005)

[147] Thiéblin, E., Haemmerlé, O., Trojahn, C.: Complex matching based on competency questions for alignment: a first sketch. In: Ontology Matching: OM-2018: Proceedings of the ISWC Workshop. p. 66 (2018)

[148] Trojahn, C., Moraes, M., Quaresma, P., Vieira, R.: A cooperative approach for composite ontology mapping. In: Journal on data semantics X, pp. 237–263. Springer (2008)

[149] Vennesland, A., Aalberg, T.: False-positive reduction in ontology matching based on concepts' domain similarity. In: International Conference on Theory and Practice of Digital Libraries. pp. 344–348. Springer (2018)

[150] Vennesland, A., Gorman, J., Wilson, S., Neumayr, B., Schuetz, C.: Automated compliance verification in ATM using principles from ontology matching. In: International Conference on Knowledge Engineering and Ontology Development. SCITEPRESS (2018)

[151] Vennesland, A., Neumayr, B., Schuetz, C., Savulov, A.: D1.1 Experimental ontology modules formalising concept definition of ATM data. Tech. rep., The BEST Project (2017)

[152] Vennesland, A., Schuetz, C., Keller, R.M., Neumayr, B., Gringinger, E.: Ontologies in air traffic management: A comparison of the AIRM and NASA ATM ontologies. In: Proceedings from the International Semantic Web Conference 2019. Springer (2019)

[153] Visser, P.R., Jones, D.M., Bench-Capon, T.J., Shave, M.J.: Assessing heterogeneity by classifying ontology mismatches. In: Proceedings of the FOIS. vol. 98 (1998)

[154] Visser, P.R., Jones, D.M., Bench-Capon, T.J., Shave, M.: An analysis of ontology mismatches; heterogeneity versus interoperability. In: AAAI 1997 Spring Symposium on Ontological Engineering, Stanford CA., USA. pp. 164–72 (1997)

[155] Vrandecic, D.: Ontology Evaluation. In: Handbook on Ontologies, pp. 293–313. Springer, Berlin, Heidelberg, 2nd editio edn. (2009)

[156] Winkler, W.E.: The state of record linkage and current research problems. In: Statistical Research Division, US Census Bureau. Citeseer (1999)

[157] Winston, M.E., Chaffin, R., Herrmann, D.: A Taxonomy of Part-Whole Relations. Cognitive Science 11(4), 417–444 (1987)

[158] Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: Proceedings of the 32nd annual meeting on Association for Computational Linguistics. pp. 133–138. Association for Computational Linguistics (1994)

[159] Yatskevich, M., Giunchiglia, F.: Element level semantic matching using WordNet. In: Meaning Coordination and Negotiation Workshop, ISWC (2004)

[160] Yu, Z., Wang, H., Lin, X., Wang, M.: Learning term embeddings for hypernymy identification. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)

[161] Zhang, Y., Wang, X., Lai, S., He, S., Liu, K., Zhao, J., Lv, X.: Ontology matching with word embeddings. In: Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data, pp. 34–45. Springer (2014)

[162] Zhou, L., Cheatham, M., Krisnadhi, A., Hitzler, P.: A complex alignment benchmark: Geolink dataset. In: International Semantic Web Conference. pp. 273–288. Springer (2018)

[163] Zhou, L., Thiéblin, É., Cheatham, M., Faria, D., Pesquita, C., dos Santos, C.T., Zamazal, O.: Towards evaluating complex ontology alignments. Knowledge Eng. Review 35, e21 (2020)