University of California
Santa Barbara

A Data-Driven Framework for Assisting Geo-Ontology Engineering Using a Discrepancy Index

A Thesis submitted in partial satisfaction
of the requirements for the degree

Master of Arts
in
Geography

by

Bo Yan

Committee in charge:

Professor Krzysztof Janowicz, Chair
Professor Werner Kuhn
Professor Emerita Helen Couclelis

June 2016

The Thesis of Bo Yan is approved.

Professor Werner Kuhn

Professor Emerita Helen Couclelis

Professor Krzysztof Janowicz, Committee Chair

May 2016

A Data-Driven Framework for Assisting Geo-Ontology Engineering Using a Discrepancy Index

by

Bo Yan

Acknowledgements

I would like to thank the members of my committee for their guidance and patience in the face of obstacles over the course of my research. I would like to thank my advisor, Krzysztof Janowicz, for his invaluable input on my work. Without his help and encouragement, I would not have been able to find the light at the end of the tunnel during the last stage of the work. Because he provided insight that helped me think out of the box. There is no better advisor. I would like to thank Yingjie Hu who has offered me numerous feedback, suggestions and inspirations on my thesis topic. I would like to thank all my other intelligent colleagues in the STKO lab and the Geography Department – those who have moved on and started anew, those who are still in the quagmire, and those who have just begun – for their support and friendship. Last, but most importantly, I would like to thank my parents for their unconditional love. While being physically apart, they have been providing their strongest support for me to thrive. None of this would have been possible without them.

Abstract

A Data-Driven Framework for Assisting Geo-Ontology Engineering Using a Discrepancy Index

by

Bo Yan

Geo-ontologies play significant roles in formalizing concepts and relationships in geography as well as in fostering publication, retrieval, reuse, and integration of geographic data within and across domains. The status quo of geo-ontology engineering is that a group of domain experts collaboratively formalize the key concepts and their relationships. On one hand this centralized top-down ontology engineering approach can take into account invaluable expert knowledge and capture our perception of the world correctly in most cases; on the other it might yield biased geo-ontologies and misrepresent some important concepts or the interplay between different concepts due to the fact that such top-down ontology engineering strategy hardly takes into consideration the existing dataset. With an increasing number of Linked Data on the Web, we are able to use such data to assist the traditional geo-ontology engineering process. However, the quality of Linked Data also imposes challenges to this task. This research proposes a framework by modeling the hierarchical structure using a series of data mining algorithms and eventually quantifies the difference between the original ontology and the data mining one with the proposed *Discrepancy Index*. The *Discrepancy Index* can help geo-ontology engineers identify as well as quantify potential ontological modeling issues and Linked Data quality issues, thus closing the gap in the dynamic process of geo-ontology engineering.

Contents

Chapter 1

Introduction

Ontology, originally a philosophical term dealing with the nature of being, has been borrowed by information scientists as a means of computationally modelling a particular domain of interest in knowledge representation tasks. It has since become one of the pillars of the Semantic Web and has been applied to various domain areas. This chapter discusses the context and motivation upon which the research is based, describes the status quo of geo-ontology in particular, and raises research questions to be answered.

1.1 Context and Motivation

In 2001, an article written by Tim Berners-Lee, James Hendler and Ora Lassila was published in the Scientific American magazine, envisioning an evolutionary transformation from the existing Web to the Semantic Web. Semantic Web, as the name implies, is the Web that has meaning. Unlike the traditional Web of Documents, Semantic Web is a Web of Data in that everything, even conceptual things, can be stored, accessed and utilized, fostering data exchange across diverse domains and enabling new types of applications on the Web.

This simple but innovative idea seemed far-fetched at that time, just like what it was when Tim Berners-Lee first coined the term World Wide Web or simply known as the Web. But in the short 25 years of its invention, the Web has become ubiquitous and indispensable in our daily life. No one could ever imagine a world without the Web. Although it's less likely that the development speed of Semantic Web is able to be comparable to that of its predecessor, which grew exponentially within the first few years, as of now, a significant fraction of the pages on the Web contain Semantic Web markup [1]. For instance, Knowledge Graph is a service provided by Google that is empowered by Semantic Web technologies. The Knowledge Graph provides users with structured and detailed information in addition to the search results on certain topics, powering richer and more interactive user experience. It is rooted in public sources such as CIA World Factbook, Freebase and Wikipedia, containing more than 500 million objects, and more than 3.5 billion facts about and relationships between these different objects [2]. Other services such as Microsoft Bing has also integrated Semantic Web technologies into their search engines, further exploiting the potential of the next generation web. The Semantic Web markup is now of the same order of magnitude as the Web itself [3].

The underpinning technologies that make this transition possible include Linked Data and ontologies. In order to realize the Semantic Web or the Web of Data, a huge amount of data as well as their relationships need to be readily available in a structured format in order to be read and managed by Semantic Web applications. This collection of interrelated datasets on the Web are referred to as Linked Data [4]. An example of the dataset is DBpedia, which is a structured version of Wikipedia. DBpedia also integrates links to other Linked Data on the Web, acting as a huge data silo. Each dataset in the Linked Data cloud has its own vocabularies to define concepts and relationships in an area of concern. These vocabularies are essentially what we refer to as ontologies. Ontology, from an information science perspective, is a formal representation of concepts and their

relationships within a domain. It is a form of knowledge representation, which provides reasoning support within a domain as well as vocabulary support for knowledge-sharing among different domains. Thus, ontology has become a core technology to foster the publication, retrieval, reuse, and integration of data within and across domains.

Ontology has played a significant role in the development of Linked Data and Semantic Web. To make this statement more convincing, take Schema.org ontology as an example. Schema.org is a collaborative, community activity sponsored by Google, Microsoft, Yahoo and Yandex with a mission to create, maintain, and promote schemas for structured data on the Internet [5]. It provides a single vocabulary across a wide range of domains. The benefit of this initiative is that it makes sure the vocabulary can be widely used and is consistent across the Web, thus webmasters are more willing to add Schema.org markups in their webpages. Many well-known websites such as nytimes.com, bbc.co.uk, imdb.com, youtube.com, and yelp.com use Schema.org ontology. The adoption rate of Schema.org ontology is 31.3% in a sample of 10 billion pages from a combination of the Google index and Web Data Commons according to [3]. This has a profound impact on the exposure and the publicity of structured data on the Web in general, further enhancing the development of Semantic Web.

1.2 Geo-ontology

From a domain scientist's perspective, ontology is also important to Geographic Information Science (GIScience). Due to the diverse and eclectic nature of geographic information, its data usually comes from different sources and formats. This heterogeneity creates a barrier for GIScientists to take full advantage of the data to do more comprehensive spatial analysis. Geo-ontology provides a way to flexibly integrate geographic information based on its semantics regardless of its representation. Another

inherent feature that distinguishes GIScience or geography in general from other disciplines is that many geographic concepts are vague and ambiguous. For instance, the concept *place* is subjective, cognitive, culturally situated, and often time-variant [6]. The concept *town* has no formalized definition in terms of area or population size because cultural and historical aspects may play a role, for example, Bloomsburg is the only town in Pennsylvania. While geo-ontology is not able to solve the vagueness of geographic concepts, the Open World Assumption provides a way to deal with the vagueness to a certain degree. The Open World Assumption embraces the idea that no single agent has a complete knowledge of the world, therefore each observer can have their own interpretation of concepts and the corresponding relationships. With this assumption, geo-ontology is able to model the geographic world based on the interpretation of domain experts. This approach of formalization fosters integration of geographic data. For these reasons, geo-ontologies have been active research areas over the last 20 years and the geospatial semantics community has been among the early adopters of the Semantic Web, contributing methods, ontologies, use cases, and datasets [7].

The status quo of the design pattern of geo-ontology and, in fact, all other ontologies is that a handful of domain experts gather together to decide on the important concepts and relationships for the domain and the task in which the ontology will be applied. Competency questions are usually raised as starting points. Discussions around these questions stimulate the formation of axioms in the ontology. This top-down practice is efficient and proven to be sufficient for usage in most cases. The challenge with this engineering artifact is that the ontology created by the top-down approach has a life cycle and soon becomes obsolete as the actual dataset or use case that populates the ontology is evolving constantly, then a new ontology design pattern will be needed to revise the original ontology. This centralized, expert-dominated way of designing ontology is labor intensive. Some Linked Data providers such as DBpedia has offered another way to

decentralize the ontology design. It allows the community to edit the ontology classes and properties if any community member finds a discrepancy between the ontology and the dataset or simply prefers a particular class or property to another. The issue with this practice is that this ontology editor from the community might be able to spot a particular problem in the ontology, but he/she might not be familiar with the good designing practices of ontology, thus it might be hard for him/her to use the ontology designing tools supported by DBpedia, making the ontology editing practice meaningless. Moreover, as typical users are not well-trained to have a holistic view of the entire ontology, especially a cross-domain ontology like DBpedia ontology, more often than not, they are just adding noise to the original ontology, making the situation even worse.

One promising way to alleviate the situation is to implement a data-driven approach. By taking a closer look at the dataset, we are able to detect the gap between the real world scenario and the corresponding ontology. This approach is especially helpful for domains that have many inherently fuzzy and ambiguous concepts and relationships, e.g. geography.

1.3 Research Questions

Ontology engineering is a dynamic process, namely it requires constant revision and refinement based on the dynamics of knowledge within and across domains. Bottom-up approaches are frequently implemented to change the common top-down ontology engineering status quo, such as using Linked Data to learn new geographic concepts and enrich the top-down geo-ontologies [8]. This research is inspired by such school of thought [9] as well as some observations on the current DBpedia geo-ontology and its corresponding Linked Dataset.

In the current version (DBpedia 2015-10) of DBpedia ontology, the class *Canal* is clas-

sified as a sibling class of *River*, which makes them both the subclass of *BodyOfWater*. This seems to be a rational classification at first glance since canals are usually channels of water. However, *BodyOfWater* is a subclass of *NaturalPlace*. Because of the transitivity of *rdfs:subClassOf* relationship, this makes *Canal* the subclass of *NaturalPlace*. The definition of *Canal* from Oxford dictionary will be helpful if this misclassification is still not conspicuous. In Oxford dictionary, *Canal* is defined as "an artificial waterway constructed to allow the passage of boats or ships inland or to convey water for irrigation". The question then lies in finding the correct classification of the class *Canal*. The definition gives us some hint. The word "artificial" and "constructed" make it a man-made geographic feature that could be group into the *Infrastructure* category. The word "waterway", "passage" and "convey" make it a *RouteOfTransportation*. By closely examining the classes in the DBpedia geo-ontology, we are able to find that *RouteOfTransportation* has a subclass *WaterwayTunnel*. *Canal* should be the sibling class of *WaterwayTunnel* under *Infrastructure* rather than be the sibling class of *River* under *NaturalPlace*. Apparently, this process demands quite a lot of domain knowledge and is tedious and time-consuming.

On the other hand, due to the nature of the provenance of most Linked Datasets, the quality concerns should also be incorporated into this dynamic geo-ontology engineering process. Although some best practices for Linked Data creation and reuse, such as using standard RDF tools and libraries, as well as some approaches for Linked Data validation and cleansing have been proposed [10], these approaches usually only detect and correct syntactical errors. Comparing the ontology and the corresponding Linked Dataset might lead to some insight about the intended meaning behind the data. Again, take DBpedia Linked Dataset as an example. Much of the structured information in DBpedia dataset comes from Wikipedia infoboxes. Direct transformation of these infoboxes into structure data results in noisy data and inconsistent nomenclature of the properties. The

DBpedia community is well aware of this issue, so with the DBpedia 3.2 release, they adopted a new infobox extraction approach that took into consideration a newly created DBpedia ontology. In this case, DBpedia can provide users with much cleaner and better structured mapping-based datasets. For instance, instead of using both *dbp:headquarter* and *dbp:headquarters* because they are used interchangeably in the original Wikipedia infoboxes, in the mapping-based dataset, DBpedia uses *dbo:headquarter* to unify these semantically identical but syntactically different properties based on the manually predefined properties in DBpedia ontology. This canonical example illustrates how data could be cleansed and refined by ontologies. Similar to the previous example, this hand-generated mapping from raw Wikipedia infobox templates to predefined ontologies is an ad hoc strategy that is hard to be generalized. It is also important to note that, although this strategy provides cleaner data, it does not provide a complete coverage of the original properties and types from the infoboxes, thus introducing data bias if users only use the mapping-based dataset.

The above mentioned two examples originate from two common issues in the course of geo-ontology engineering or ontology engineering in general: ontological modeling issue and Linked Data quality issue. Conventional approaches treat these two issues separately and on a case-by-case basis because experts in tackling these two issues do not speak the same language and the subtle interplay of these issues has not been well-recognized. In an effort to solve this problem and provide ontology engineers with more general approaches, in this research, we aim to find answers to the following questions.

Research Questions: How can we discover the mismatch between a particular geo-ontology and its corresponding Linked Dataset? How can we quantify the mismatch? Based upon the mismatch, what can we learn in terms of ontological modeling and Linked Data quality?

As the research tasks point out, this work attempts to solve the problem in two

directions. One direction is to investigate the role of Linked Dataset in informing and detecting potential geo-ontology modeling issues (such as misclassification); the other is to identify and highlight the role of geo-ontology in helping ontology engineers discover potential Linked Data quality issues (such as nomenclature issues and data bias). We realize that these two components are integral parts of geo-ontology engineering and should be incorporated into the process as a whole. In light of this, we also propose a general framework/workflow for geo-ontology engineers to follow. The mismatch here is a relatively neutral term because the cause of the mismatch could either be an ontological modeling issue or a Linked Data quality issue. Geo-ontology engineers should be aware of both possibilities and make decisions accordingly. If it's an ontological modeling issue, geo-ontology engineers can revise the geo-ontology; if it's a Linked Data quality issue, data cleansing can be applied to remedy the issue.

The primary contribution of this research lies in finding measurements that help identify and quantify the difference in geo-ontology and the corresponding Linked Dataset within a systematic framework, so that ontology engineers can further look into the issues concerning both ontological modeling and Linked Data quality more easily and more efficiently. The scope of this research, in terms of domain areas, is limited to the field of geography while this research framework and workflow can be easily applied to a variety of other domains of expertise; in terms of dataset, is limited to DBpedia because it contains a large amount of geographic data and its ontology and instances that filled the ontology are publicly accessible, but likewise, it can be applied to other Linked Datasets.

1.4 Methods

Numerous methods have been used in collecting and analyzing the data and ontology. This section outlines the main methods that are employed.

While some simple statistics can be used to describe the data, it is usually better to visually present the information. In this case, a network visualization is adopted to illustrate the hierarchical relationships of the ontology. The relative size of the nodes represents the number of instances in each class while the edge represents the superclass-subclass relationships. The network makes it easier to identify the community clusters and the hubs in the ontology.

The main part of this framework is an unsupervised learning task. It starts from choosing suitable feature spaces. In order to reduce the dimensionality of the data, Multidimensional Scaling (MDS) is adopted. To build the concept hierarchy based on the instance data in DBpedia, hierarchical clustering algorithm is implemented. The advantage of hierarchical clustering is that the cluster number does not need to be specified and it's suitable for hierarchically structured dataset. The derived hierarchical structure will be compared with the original ontology hierarchy. The is done by adopting the semantic similarity measure proposed in [11]. This semantic similarity measure is exploited as the basis of the similarity measures for different concepts within the ontology. The detailed explanation is presented in the corresponding chapter.

1.5 Outline

The remainder of this thesis is outlined and summarized as follows. Chapter 2 reviews some related work and unsolved issues on the data-driven ontology methods, pointing out the importance of this research. Chapter 3 introduces the workflow and explains in detail the methodologies proposed in the research. Chapter 4 presents the experiment using DBpedia geo-ontology and Linked Dataset. Chapter 5 gives conclusions about the research and provides direction for future work.

Chapter 2

Related Work

This chapter reviews the existing research on ontology learning and ontology evaluation. Approaches for ontology learning can be classified into three categories: ontology learning from unstructured data, ontology learning from semi-structured data, and ontology learning from structured data. While unstructured data is the most challenging source of ontology learning, structured data gives researcher more hint on how the ontology should be like. Linguistic methods, machine learning, data mining, and an integration of them are exploited for all three categories. Ontology evaluation can be categorized as gold standard-based evaluation, user-based evaluation, application-based evaluation, and data-driven evaluation.

2.1 Approaches for Ontology Learning

Ontology learning, coined by Maedche and Staab, aims at facilitating the construction of ontologies using unstructured, semi-structured and fully structured data [12]. This automatic or semi-automatic ontology engineering process includes extraction, refinement and evaluation. Normally, ontology construction is a tedious and time-consuming task

in that it requires an in-depth understanding of the domain and domain experts have to come to terms with each other and agree upon different interpretations of certain concepts. In most cases, the results could be incomplete and inaccurate [13]. Ontology learning is a way of dealing with these problems.

During the last decade, researchers have proposed several ontology learning methods and frameworks to help engineers in building the ontology. The methods cover a wide range of fields such as natural language processing, artificial intelligence, machine learning, information retrieval and knowledge acquisition [13]. There are many ways to categorize ontology learning approaches. Many works tend to summarize ontology learning techniques based on the types of input from which the ontology is constructed [13, 14, 15, 16] This review follows this categorization. There are three types of input from which the ontology can be learned and all these data types have been studied and the corresponding methods proposed. These three types are unstructured data input, semi-structured data input and structured data input. Examples of unstructured data include free text. Semi-structured data include sources that have predefined structure such as XML files. Structured data includes databases, existing ontologies, knowledge bases, and dictionaries.

2.1.1 Ontology Learning from Unstructured Data

Unstructured data is the most difficult type of input and needs an extra layer of processing because of the noise and the lack of structure. The unstructured data usually comes from natural language text extracted from a corpus or a web content. Many of the methods in ontology learning from unstructured data are inspired by previous work in computational linguistics [16]. Approaches using unstructured data to learn ontology frequently take advantage of statistical analysis, natural language processing techniques

and the combination of both [13].

Aussenac-Gilles et al. [17] propose a methodological framework independent of the language used in text. Their method consists of four levels: corpus constitution, linguistic analysis, normalization and formalization. In the corpus constitution step, designers select the available technical documentations based on the requirements and description of the application. Then the corpus is digitized so that natural language processing tools can be applied to select the terms and lexical relations during the linguistic analysis stage. Since the result of the linguistic analysis is still coarse, normalization is necessary. In the normalization step, designers have to manually filter out less important terms and relations based on their significance both in the domain and for the application. After the refinement, a semantic network of hierarchical relations is composed by normalizing the terms and relations into concepts and semantic relations. During the final phase, semantic concepts and relations are formalized and inserted in the ontology. The disadvantage of this framework is that in the linguistic analysis part and the normalization part, the selection of important terms and relations are subjective and largely dependent on the designer?s interpretation of the domain and application description. The whole process is very tedious and time-consuming, thus it?s not an efficient and cost-effective way to learn ontologies from text.

Cimiano et al. [18] propose an automatic approach to acquire concept hierarchies from a text corpus using Formal Concept Analysis (FCA). FCA is a method used to derive implicit relationships, e.g. concept hierarchies, from a collection of objects and their properties [19]. The linguistic analysis is implemented at the first stage. A linguistic parser is used to acquire the context of a certain terms from the corpus based on distributional hypothesis described in [20]. After lemmatization and smoothing, FCA is applied to produce a lattice and the lattice is transformed into a partial order constituting a concept hierarchy. The evaluation result shows that the FCA-based approach is

comparable to, or even better than, similarity-based clustering methods [18]. The disadvantage of this method is that the lattice size is likely to increase exponentially which leads to an exponential time complexity.

Khan et al. [21] present a mechanism to construct ontologies automatically from text using clustering approaches. A modified Self-Organizing Tree Algorithm (SOTA) [22] is proposed and WordNet [23] is used for identifying concepts and sub-concepts. The approach can be split into two steps. The first step is to use the clustering algorithm, the modified SOTA to cluster a set of documents in to a multi-level hierarchy. The second step is to assign a concept from WordNet for each node in the hierarchy. The classical Rocchio algorithm [24], supervised Self-Organizing Maps (SOM) algorithm [25] and cosine similarity measures are used to determine the topic for each document. For keywords associated with multiple concepts, concept sense disambiguation is performed using cosine similarity technique. Their evaluation shows that the modified SOTA renders better results than Hierarchical Agglomerative Clustering (HAC) [26].

2.1.2 Ontology Learning from Semi-Structured Data

Semi-structured data includes sources that have predefined structure, such as XML files. Ontology learning from semi-structured data frequency uses data mining techniques.

Papatheodorou et al. [27] describe a data mining process that adopts the Cluster Mining approach [28] to construct hierarchies from XML/RDF metadata. The entire data mining process includes data collection and pre-processing, pattern discovery, and pattern post-processing and evaluation. The objective of the data collection and pre-processing part is to select the appropriate keywords from the metadata that facilitates the discovery of similarities among documents. For this purpose, stop words are filtered out. Then WordNet [23] is used to extract word roots. During the pattern discovery

stage, the Cluster Mining approach is used to discover and build clusters of similar documents. In order to examine the clusters from the last stage, they use some simple statistical indicators, such as the difference between the keyword frequency in the entire dataset and the cluster, to measure descriptiveness of the clusters.

Karoui et al. [29] use HTML pages to extract ontologies with the assistance of clustering methods. Initially, they exploit the HTML page tags to generate concepts. Then they apply approaches described in [17] to generate candidate terms as well as normalize and formalize them. Next, they build a data table whose fields contain the word, the labeled word, the grammatical type of the word (noun, adjective, etc.), the style of the word (title, bold, etc.), a number representing how many times the style appears in the document and the number of document that locate the word. They use this table and the help of a divisive clustering method to generate a hierarchy of clusters of words. Concepts are generated by the user based on the clusters of words.

Zarrad et al. [30] aim to extract the taxonomy of concepts in a given field by analyzing the text structure of web documents. Their approach can be divided into the following steps: corpus pretreatment, concepts extraction, and taxonomic links extraction. During the corpus pretreatment step, a Tree Tagger [31] is used to associate each word with its grammatical type (noun, adjective, etc.) and its canonical form. Then a parser is used to extract information from the tags of the HTML documents. Stop words are removed in this step. Candidate terms selected using syntactic approach (analyzing the grammatical role of words in text) and statistical approach (analyzing the frequency of the words in text). This step is similar to the preprocessing step in [27]. In the concept extraction step, they propose Corpus Relevance (CR) and Inverse Corpus Frequency (ICF) as the measurements to extract concept. CR corresponds to the relevance of a candidate term in all the documents of the corpus. ICF indicates the opposite frequency of the corpus. In the end, taxonomic links are extracted using linguistic methods, projection of lexico-

syntactic patterns, and analysis of titles' hierarchy in documents.

2.1.3 Ontology Learning from Structured Data

Structured data includes databases, existing ontologies, knowledge bases, and dictionaries. The most common one is to extract ontology from relational databases.

Johannesson [32] presents a method that translate a schema in a traditional data model into a conceptual schema. The first step of this method transform relational schemas into a form appropriate for identifying object structures. Certain cycles of inclusion dependencies are removed and certain relation schemes are split. Then the relational schema is mapped into a conceptual schema. Each relation scheme gives rise to an object type, and the inclusion dependencies give rise to either attributes or generalization constraints, depending on the structure of the keys in each relations scheme. Four different transformations have been used in this work: candidate key splitting, inclusion dependency splitting, folding, and schema mapping. The candidate key splitting is used when the relation scheme in the third normal form corresponds to several object types. The inclusion dependency splitting is applied when a single relation corresponds to several object types. Folding is used when several relation schemes correspond to a single object type. Schema mapping is applied to map a relational scheme to an object type. The first three steps are implemented repeatedly before a user decides the final conceptual schema in the schema mapping stage.

Kashyap [33] proposes an approach for designing on ontology for information retrieval based on the schemas of the databases and a collection of queries that are of interest to the users. The whole process is two-fold: the first step is to construct ontology from database schemas, the second step is to refine the constructed ontology based on user queries. The first step is essentially a database schema reverse engineering work,

which entails abstracting the local data-specific details in the underlying schemas and reverse engineering an Entity-Relation model from the underlying schema elements such as tables, columns, etc. After detail abstraction, information is grouped from multiple tables, relationships are identified, and mappings between elements of the ontology and schema are defined. After the development of the ontology from the database schema, users are expected to run some queries to help revise the ontology. The basic idea of this revision is that, some attributes, relationships and class-subclass relationships may be overlooked in the ontology construction step while become obvious on perusal of the queries.

Williams et al. [34] describe an approach for identifying candidate relations between expressive, diverse ontologies using concept cluster integration. Their approach includes several steps: concept representation and learning, concept interpretation and verification, and concept cluster integration. In the concept representation and learning stage, a concept vector is made by a vector of ones and zeroes corresponding to the presence or absence of a toke (word and HTML tag from the web page) in a web page. The agents use supervised inductive learning to learn the individual ontologies. The output of this ontology learning is semantic concept descriptions (SCD) in the form of interpretation rules. For each of these SCD rules, an associated certainty value is determined during the learning process. Agents then use concept-based queries (CBQ) to communicate their requests for concepts related to the query. After sending out CBQ, agents will receive response indicating whether the other agent knows a similar or related concept. After the interpretation and verification, agents will apply concept cluster integration algorithm to look for candidate relations between ontologies. Their approach does not identify the specific type of relationship in the ontologies but assumes they consist of general is-a (class-subclass) relations.

2.2 Approaches for Ontology Evaluation

Another strand of research concerning ontology that is related to the work in this thesis is ontology evaluation. Ontology evaluation primarily focuses on two important aspects of ontologies: correctness and quality [35]. Ontology quality emphasizes on the internal attributes of the ontology and the influences of these attributes on external quality attributes, whereas, on the ontology correctness perspective, ontology is an approximation of a domain and ontology correctness is concerned with the distance between this approximation and the real world. Recent research [35, 36, 37] classifies ontology evaluation methodologies based upon gold standard-based evaluation, user-based evaluation, application-based evaluation, and data-driven evaluation. The levels on which these above mentioned types of ontology evaluation methods can perform on includes: lexical, vocabulary, concept, and data level; hierarchy and taxonomy level; other semantic relation level; context and application level; syntactic level; and structure, architecture, and design level [35].

For the gold standard-based evaluation, the gold standard can be a well-constructed ontology or prepared by a domain expert. The gold standard evaluation of the ontology focuses on lexical, conceptual level, hierarchy level, other semantic relations level, and syntactic level. It is usually based on precision and recall measures [38]. While Maedche et al. [38] measures the similarity of ontologies on the conceptual and lexical level based on the extent to which one ontology is covered by the other, Brank et al. [39] focuses on the hierarchy and taxonomy level of the ontology as well as the instance arrangement by comparing two ontologies constructed using the same set of instance. The major downside of the gold standard methodology is that the quality of the gold standard is difficult to measure and it requires a lot of manual labor.

User-based evaluation is normally done by humans and is based upon human as-

sessment of the degree to which the ontology meets the requirements set by human?s perspective. Supekar [40] proposes a user-based evaluation method that allows the user to provide qualitative ratings on the ontology content. The method takes advantage of metadata elements and develops an ontology model that captures the quality features of an ontology. Users then can associate these elements to the ontology and evaluate the ontology. The conspicuous drawback of this type of evaluation is that user experience is relatively subjective, so the objectivity of the standards is hard to establish [35].

Application-based evaluation typically assesses the the ontology with respect to an application scenario. Porzel et al. [41] present an evaluation method that allows to measure the performance of different ontologies on specific tasks. They use a well-defined benchmark task to measure the ontology-dependent part of the performance, then test and incrementally augment the ontologies. The main issue with this type of evaluation approach is that the result of the evaluation is application or task-specific, meaning the evaluation result is heavily based on the application context and is hard to generalize.

Data-driven evaluation takes the populating data of the ontology into consideration. Brewster et al. [36] measures the fit between the ontology and a domain of knowledge based on the corpus. The method compares the ontologies with text from the corpus and implements a probabilistic approach to find the best fit for the corpus. The limitations of the type of evaluation is that currently the dynamics of the domain is not taken into account.

Chapter 3

Method

This chapter elaborates on the data mining algorithms and approaches adopted in the research. A systematic workflow is illustrated, followed by detailed explanations of the methodologies within the workflow. The data mining methods adopted in this research aim to render a hierarchical structure based upon the feature space derived from the Linked Dataset. This data mining hierarchy is compared with the original hierarchy and a *discrepancy index* is proposed for the purpose of assisting ontlogy engineers in refining the ontology.

3.1 Workflow

The objective of outlining the workflow is to provide a concise and systematic demonstration of all the procedures and algorithms implemented in this research and propose a framework for applying our approaches to other ontologies. Figure 3.1 shows an overview of the workflow in this research. Since ontology engineering is a dynamic process, this workflow illustrates that some procedures in this ontology engineering assistance system are recursive, which means that they can be applied repeatedly in order to refine the

ontology based on previous results.

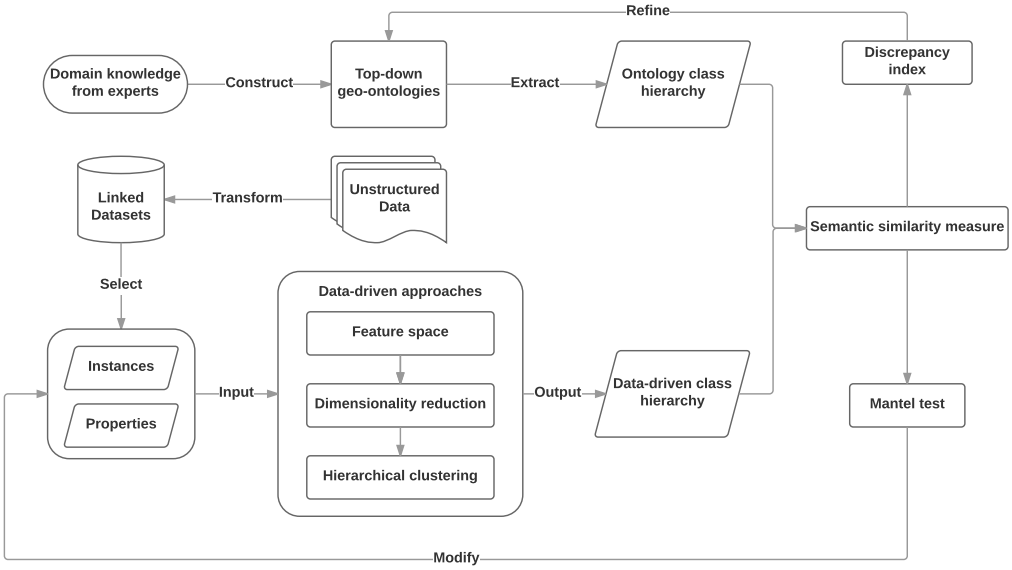

Figure 3.1: An overview of the workflow

This workflow consists of two parallel processes. The first process originates from the top-down geo-ontologies which are constructed manually by expert with their domain knowledge. The second thread of this workflow comes from Linked Datasets that are transformed from unstructured data, such as Wikipedia pages. This thread focuses on the bottom-up part.

Let's proceed on the bottom-up part first. Similar to the idea of generating features from Linked Open Data using data mining algorithms [42], from a monolithic Linked Dataset, we use SPARQL queries to select instances and properties concerning the specific classes in the top-down ontology. These instances and properties are then act as input for our three-stage data-driven approach. The first stage in the data-driven approach is to construct a raw feature space based on the input data. Because of the nature of the dataset, the raw feature space will be of high dimension and will be very sparse. We transform our raw feature space into a lower dimensional space using dimensionality reduction methods in exchange of computational efficiency and efficacy. In the last

stage of the data-driven approach, we use hierarchical clustering algorithms to obtain the hierarchies derived from the dataset. In the next step, semantic similarity measures are implemented to compute the pairwise similarity between each pair of classes in the derived ontology hierarchy. Meanwhile, the original ontology hierarchy is extracted from the top-down ontology. Semantic similarity measures are also applied on this hierarchy to obtain a similarity matrix of the original ontology. In order to evaluate whether the derived hierarchy correlates with the original one, we run a Mantel test [43] to check the significance of their correlation. If their correlation is not significant, we go back to the first stage in the data-driven approach to modify our feature space and rerun our algorithms in order to obtain a significant correlation. After we find a hierarchy derived from the feature space that is significantly correlated with the original hierarchy, we calculate the *discrepancy index*, which provides guidelines to refine the original top-down ontology. For the purpose of simplicity, we elaborate on each of the methods and algorithms in the context of DBpedia place ontology and its dataset. However, these methods and algorithms, together with the workflow and framework, can easily be applied to general use cases.

3.2 Ontology

Most Linked Datasets are bundled with their corresponding ontologies, usually in a OWL file. Because we only need the hierarchy of the classes within the ontology, we can use SPARQL queries to find the subclasses of a class recursively and extract the hierarchical structure in the ontology. Here is an example of SPARQL query to extract all the subclasses of a class:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
```

```
SELECT DISTINCT ?class
WHERE {
        ?class rdfs:subClassOf* :SomeClass .
        ?class a owl:Class .
}
```

3.3 Instance and Property Extraction

In order to apply data mining algorithms to the dataset, we have to find a feature space for our data. The features should be representative of the difference between classes and should be related with the instance data. Class property satisfies both criteria. A collection of properties in each class reveals the fundamental characteristics in that these properties act as relationship links between each class and another entity. Different classes will have different collections of properties. While a class by definition can have various properties, each instance in this class does not have to have all these properties. This variation between the class definition and the populating instance is the basis for the data mining algorithms.

With the help of the SPARQL query, we are able to retrieve the features. Let P be the set of features, so $P = \{ P_1, P_2, P_3, ..., P_n \}$. Each one of the instances is defined as a row vector $\boldsymbol{I_i} = [p_{i1} \quad p_{i2} \quad p_{i3} \quad \cdots \quad p_{in}]$ in the instance set $\boldsymbol{I}$. We use SPARQL queries to check each instance and assign values to the instance vector based on whether or not it has any of the features as described in Equation 3.1. After constructing the instance vectors, we aggregate the instances into classes, i.e. grouping instances of the same class together. Let $\boldsymbol{C}$ be the set of classes, so $\boldsymbol{C} = \{ \boldsymbol{C_1}, \boldsymbol{C_2}, \boldsymbol{C_3}, ..., \boldsymbol{C_j} \}$. If $\boldsymbol{I_i}$ is an instance of class $\boldsymbol{C_i}$, we say $\boldsymbol{I_i} \in \boldsymbol{C_i}$. We define the class vector $\boldsymbol{C_k} = [p_1^k \quad p_2^k \quad p_3^k \quad \cdots \quad p_n^k]$ as the

sum of instance vectors that belong to the class $\boldsymbol{C_k}$, as formalized in Equation 3.2.

$$p_{il} = \begin{cases} 1 & \exists P_l \\ 0 & \nexists P_l \end{cases} \tag{3.1}$$

$$p_i^k = \sum_{\boldsymbol{I_j} \in \boldsymbol{C_k}} p_{ji} \tag{3.2}$$

Using the above mentioned definitions and equations, we can convert the raw instance and property data into a set of class vectors. Each class vector has values representing the frequency of the features across all instances within the class. In order to show the relative distribution of the features, we normalize the frequency count by the total number of instances in each class. Following this process, we can obtain a feature space that is able to be used in the next step.

3.4 Dimensionality Reduction

The properties in the ontology are usually very diverse and, in most cases, particular classes will only cover a small fragment of all the properties, making the feature space sparse and high-dimensional. The potential issue with high dimensional dataset is that much of the information is redundant and sparse. It is also very hard to understand and analyze such data. Apart from these, the computational efficiency of performing data mining and machine learning algorithms on such dataset is not optimistic. The term *curse of the dimensionality*, coined by Bellman [44], refers to the fact that the size of the data needed to estimate the function to several variables to a certain degree of accuracy increases exponentially as the dimensionality increases [45]. It is practically infeasible as well as computationally implausible to obtain such tremendous amount of data and run

data mining algorithms on it. Moreover, high dimensionality indicates sparser feature space which adversely influences the efficiency and efficacy of many commonly used data organization strategies. So reducing the dimensionality of these feature set is a necessary step before diving into deeper and more complicated analysis. We choose MDS as our dimensionality reduction algorithm because it 1) preserves the distance between data points and 2) provides various loss functions for better fitting the data points.

The MDS approach takes a distance matrix as its input, so we need to convert the raw feature sets into distance matrices. Because the frequency ranges of the same feature for different classes vary greatly, normalization of the class vectors is necessary. After normalization, all features are scaled to have standard deviation one and mean zero. In order to construct the distance matrices, we use Euclidean distance $d(C_i, C_j)$ described in Equation 3.3 to calculate the pairwise distance between classes within the feature set, where n is the number of features in the respective feature set. In this way, we can obtain the distance matrix $\boldsymbol{D}$.

$$d(C_i, C_j) = \sqrt{\sum_{k=1}^{n} (p_k^i - p_k^j)^2} \tag{3.3}$$

The idea of MDS is to find configuration points $\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n} \in \mathbb{R}^k$ in such a way that the given distances $d(C_i, C_j)$ are well-approximated by the distances $\parallel \boldsymbol{x}_i - \boldsymbol{x}_j \parallel$ [46]. Practically, a loss function is adopted to measure the the goodness of fit between the given distances $d(C_i, C_j)$ and the approximation distances $\parallel \boldsymbol{x}_i - \boldsymbol{x}_j \parallel$. The goal of solving a MDS problem is to minimize the loss function. Since we are using the Euclidean distance, the method we are going to implement is the classical MDS, which uses the loss function defined in 3.4.

$$Strain_D = \sum_{i,j=1}^{n} (d^2(C_i, C_j) - \parallel \boldsymbol{x}_i - \boldsymbol{x}_j \parallel^2) \tag{3.4}$$

In order to find the number of dimensions to retain with minimum loss of information from the original feature sets, we adopts Kaiser criterion [47], namely retaining dimensions with eigenvalues greater than 1. Essentially, we only ignore dimensions that extract information less than the equivalent of one original feature. This criterion is one of the most widely adopted. We will use these results as the input for clustering algorithms.

3.5 Hierarchical Clustering

Because the class-subclass relationship is essentially a hierarchical relationship, the intuitive approach to cluster the data based on the above constructed feature space is hierarchical clustering. The advantage of using hierarchical clustering is three-fold: firstly the result is a set of nested clusters organized as a hierarchical tree so that it correspond to meaningful taxonomies; secondly this clustering algorithm does not require the specification of the number of clusters; finally the clustering result can be visualized in a dendrogram (a tree based representation) and can be compared with the original ontology network visualization.

The hierarchical clustering algorithm described in this research refers to the agglomerative hierarchical clustering, which is the most popular one. The hierarchical clustering algorithm [48] starts by defining a similarity/dissimilarity measure between each pair of observations, which can be calculated using Equation 3.3. Each of the n classes is regarded as its own cluster in the beginning. The hierarchical clustering algorithm continues by joining the two clusters that are most similar iteratively at each stage until only a single cluster remains.

After constructing the distance matrices, we need to choose the linkage for our hierarchical clustering algorithm. A linkage is defined as the disimilarity between two groups of classes. The most common types of linkage include: single linkage, complete linkage,

average linkage and Ward's method. Here we use Ward's method as the linkage for calculating the distance between two clusters because it has the advantage of being less susceptible to noise and outliers and it is appropriate for quantitative variables. Ward's method defines the distance/disimilarity between two clusters as the extent to which the sum of squared distance will increase if we merge them. So it is a method that aims to minimize the increase in total intra-cluster variance after merging. Whether or not two clusters should be merged is determined by the merging cost Δ defined in Equation 3.5,

$$\begin{aligned}\Delta(A,B) &= \sum_{i\in A\cup B} \|\vec{x}_i - \vec{m}_{A\cup B}\|^2 - \sum_{i\in A} \|\vec{x}_i - \vec{m}_A\|^2 - \sum_{i\in B} \|\vec{x}_i - \vec{m}_B\|^2 \\ &= \frac{n_A n_B}{n_A + n_B} \|\vec{m}_A - \vec{m}_B\|^2\end{aligned} \tag{3.5}$$

where $\vec{m}_i$ is the center of cluster i and n_i is the number of points in the cluster i. Δ is called the merging cost of combining cluster A and cluster B. It reveals the trade off between separation and balance. For clusters that are the same distance away, smaller ones will be merged according to this formula.

3.6 Validation

At this point, we have successfully constructed the hierarchical structure based on the feature space extracted from the Linked Dataset. We need to verify whether the derived hierarchy correlates with the original ontology before proceeding to the next step. In order to achieve this task, we divide it into two subtasks. First, we compute the pairwise semantic similarity between classes within the same hierarchy with regard to the structure of the hierarchy; second, we use Mantel test [43] to test the correlation between hierarchies using the similarity matrices.

Similarity measures originate from psychological studies on determining and under-

standing how individuals are grouped into categories [49]. Semantic similarity measures have been developed under the influence as well as giving rise to the existence of structured knowledge representations offered by ontologies [50]. Previous work has been investigating semantic similarity measures as quality indicators in ontology engineering [51]. Semantic similarity measurements can be classified into four main categories: structure-based measures, information content-based measures, feature-based measures, and hybrid measures.

In this work, we choose an information content-based semantic similarity measure proposed by Jiang et al. [11]. Jiang uses semantic distance to obtain semantic similarity:

$$Dist(C_1, C_2) = IC(C_1) + IC(C_2) - 2{\times}IC(LSuper(C_1, C_2)) \tag{3.6}$$

where $IC(C_1)$ and $IC(C_2)$ are the information content of class C_1 and class C_2 respectively. $LSuper(C_1, C_2)$ denotes the most specific super class of class C_1 and class C_2.

The information content is proposed in [52] and defined by Equation 3.7.

$$IC(C) = -\log(\frac{\frac{|leaves(C)|}{|subsumers(C)|} + 1}{max\_leaves + 1}) \tag{3.7}$$

where $|leaves(C)|$ is the number of leave classes corresponding to class C, $|subsumers(C)|$ is the number of superclasses (including class C) of class C, and max_leaves represents the number of leaves with respect to the root class of the hierarchy.

In order to get the similarity, we use the inverse distance measure in Equation 3.8. We use $Dist(C_1, C_2) + 1$ to avoid 0 in the denominator. In this case, similarity values are in range $[0, 1]$. After computing the pairwise similarities between concept classes within the same hierarchy, we construct similarity matrices for each hierarchy. These similarity

matrices will be the input for the Mantel test.

$$Sim(C_1, C_2) = 1/(Dist(C_1, C_2) + 1) \tag{3.8}$$

In order to examine the correlation between the derived hierarchical structures and the original one, we will need to apply regression models on them. However, doing a single regression will violate the notion of independence since we computed the pairwise similarities for each hierarchy. We will need to randomize our data and run Monte-Carlo tests to compare the regression slope from our data with randomized results to check whether it is significant or not. In order to do this, we implemented the Mantel test.

The Mantel test is a method that has been widely used to test the linear or monotonic independence of the elements in two proximity matrices [53]. Our null hypothesis for the Mantel test is that the pairwise semantic similarity between concept classes in the man made ontology is unrelated with the pairwise semantic similarity bewteen concept classes derived from our data-driven approaches. If the p-value is less than 0.05, there is a strong evidence against our null hypothesis, meaning the derived hierarchy correlates with the original hierarchy. We can continue our next steps in this case, otherwise we need to go back to our initial selection of feature space step and redo the whole data mining process until we find a feature space that can yield a hierarchical structure that is well-correlated with the original one.

3.7 Discrepancy Index

In many scientific disciplines including geography, we tend to adopt descriptive statistics to make sense of and summarize the dataset. For example, data scientists use mean and median to describe the central tendency as well as variance and standard deviation

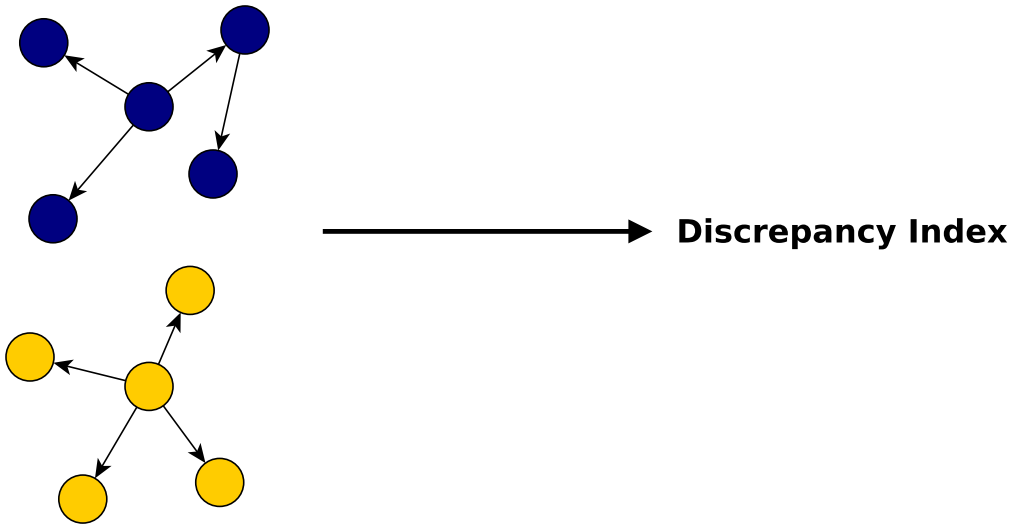

Figure 3.2: An overview of the *Discrepancy Index*

to describe the variability and dispersion of the given dataset. Geographers use Moran's I to measure spatial autocorrelation. In demography, researchers use health indicators to quantify the characteristics of a population in order to describe the health of a population. Following this train of thought, we would like to devise an index that can summarize the implicit mismatch between the geo-ontology and the corresponding Linked Dataset. In this case, we can abstract away from the intricate hierarchy of the geo-ontology as well as the large volume of Linked Dataset (see Figure 3.2). This index will also provide potential criteria basis for geo-ontology or geospatial Linked Data consumers to choose the most suitable ontology or Linked Dataset for their needs.

We propose a Discrepancy Index that can be used to assist ontology engineers to quantify the difference between the ontology and the dataset. Equation 3.9 shows the definition of Discrepancy Index Matrix. This matrix gives us all the combinations of Discrepancy Indices for the ontology and dataset. In the definition below, $SimMatrix_{original}$ is the similarity matrix representing the pairwise semantic similarity calculated from the hierarchical structure in the original ontology, likewise, $SimMatrix_{derived}$ is the similarity

matrix for the derived hierarchy. By subtracting $SimMatrix_{derived}$ from $SimMatrix_{original}$, each element in the resulting matrix $IndexMatrix$ represents the difference between the similarity of each pair of classes in the original ontology hierarchy and the derived one.

$$IndexMatrix = SimMatrix_{original} - SimMatrix_{derived} \tag{3.9}$$

The value range for each element in $IndexMatrix$, that is the Discrepancy Indices, is $[-1, 1]$. If $IndexMatrix(i, j) > 0$, it implies that class i and j are less similar in the derived hierarchy; whereas if $IndexMatrix(i, j) < 0$, it tells us that class i and j are more similar in the derived ontology. The value of $|IndexMatrix(i, j)|$ gives us the information about the extent to which the similarity in two hierarchies differ from each other. This Discrepancy Index is useful in assisting ontology engineers to refine and further develop the ontology in that it gives guidance on correcting the potential modeling bias (such as misclassification) and/or spotting the intrinsic data bias (such as missing data).

Chapter 4

Experiment

This chapter presents the experiment conducted in this research using DBpedia geo-ontology and its corresponding Linked Dataset. After applying the workflow, we revisit our observations described in the introduction chapter and examine the Discrepancy Index between *Canal* and *River*. We also provide two more examples in this experiment to showcase the usefulness of our methods.

4.1 Data Source

Among the various different types of ontologies, we narrow down into three candidate ontologies — DBpedia ontology, Schema.org ontology and GeoNames ontology. In the end, we choose DBpedia place ontology as our base ontology and its corresponding data as our data source. Schema.org ontology is a widely-used cross-domain ontology that is primarily managed and maintained by big commercial IT companies such as Google and Microsoft. GeoNames ontology is a domain-specific ontology. Although the first two are cross-main ontologies, they provides a specific place class that covers geo-ontologies. Schema.org ontology does not natively come with an OWL (Web Ontology Language)

formatted ontology, because it's primarily used in enhancing search results on the web, it uses RDFa format which is a markup for HTML pages. The data from which Schema.org ontology is extracted originated from multiple sources and millions of webpages, which makes obtaining and analyzing the data less practical and time-consuming. GeoNames ontology specializes on geospatial semantic information and the ontology is extracted from the gazetteer files which is organized by countries, resulting in a shallow hierarchy of the concepts in the ontology. This is a relatively distinct organization of dataset comparing with DBpedia and Schema.org. DBpedia is a multilingual cross-domain knowledge base that has various classes clustered into a hierarchical order and numerous properties describing the classes and their instances [54]. As shown in Figure 4.1 DBpedia is also the hub of the Linked Data cloud as of August 2014 [55]. In light of the above comparison, we choose a subset of DBpedia ontology as it provides the standard OWL formatted ontology and readily available data dumps.

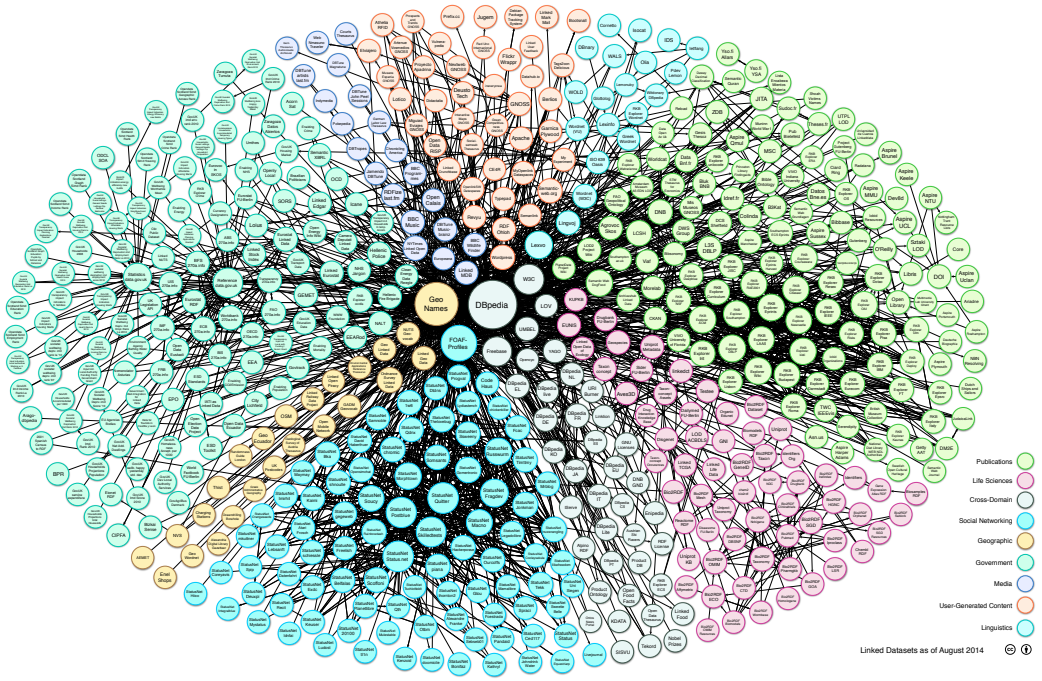

Figure 4.1: The Linked Data cloud network visualization as of August 2014

4.2 Feature Selection

Since DBpedia ontology is manually created based on the infoboxes in Wikipedia, the changes in Wikipedia pages will be reflected in DBpedia ontology. This intrinsic dynamics of ontologies is called Ontology Evolution [56].DBpedia provides these snapshots that reflect the timely adaptation of ontology to the changes in the domain. These snapshots are provided as releases by the DBpedia community. The community endeavor has refined and enhanced the ontology across different releases. We assume the ontology has become cleaner in the newest release in spite of the fact that the number of classes in the ontology has increased dramatically (see Figure 4.2), so we choose the newest release DBpedia 2015-10 ontology. DBpedia 2015-10 ontology, referred to as DBpedia ontology in the rest of this thesis, has 739 classes in total and 168 of which are subclasses of place.

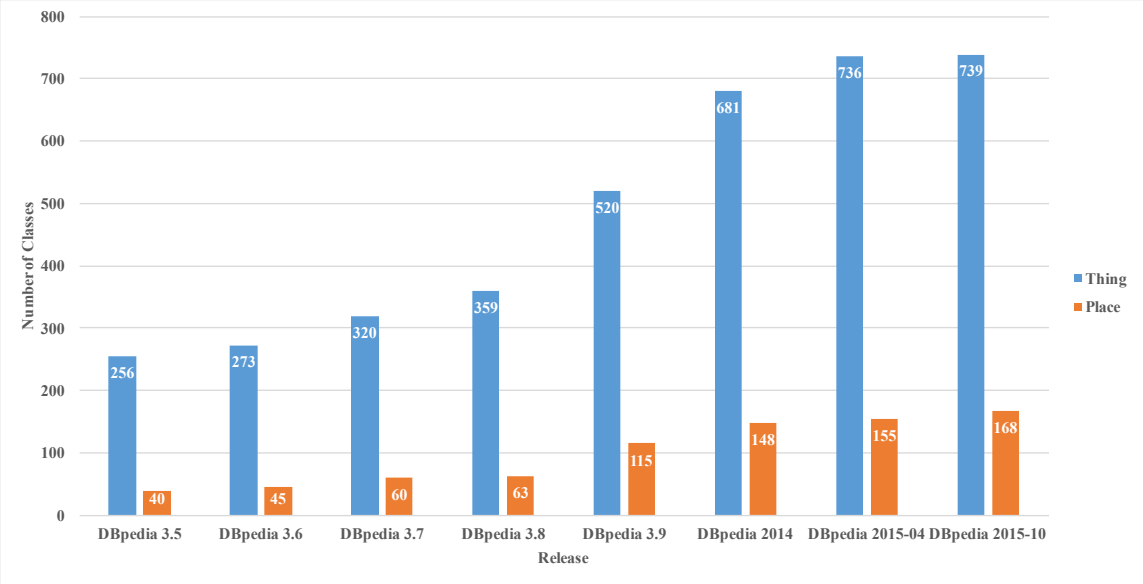

Figure 4.2: The number of classes in different releases of DBpedia ontology

The OWL file provided by DBpedia contains the entire hierarchy of the ontology, of which geo-ontology is only a portion, so we need to extract geo-ontology from the entire

ontology manually. This is accomplished by executing SPARQL queries on DBpedia endpoint with the support of Apache Jena Java libraries. Since we need the entire hierarchy of the class-subclass relationships in the place class, the transitivity feature of the property rdfs:subClassOf is taken advantage of. The following SPARQL query is used to extract geo-ontology classes from DBpedia ontology.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX dbo: <http://dbpedia.org/ontology/Place>
SELECT DISTINCT ?class
WHERE {
        ?class rdfs:subClassOf* dbo:Place .
        ?class a owl:Class .
}
```

The query will only return a flat list of all the subclasses of the place class. In order to obtain a hierarchical clustering of the ontology, each class should be queried individually about its subclasses. The final result of the class hierarchy is visualized as a network (see Figure 4.3). Each node represents a class and each edge represents the class-subclass relationship between two classes with the direction from subclass to superclass. The size of the node is proportional to the number of the instances within each class. The color scheme helps to differentiate between classes on the second level of the hierarchy, meaning the direct subclasses of place. Classes with no instances are represented in faded color.

In the feature extraction stage, we focus on features based on properties in each class. The properties here in DBpedia Linked Dataset are analogous to attributes in different place types. The rationale behind it is that similar place types share similar attributes while distinct place types have distinct attributes. For example, place type *City* and

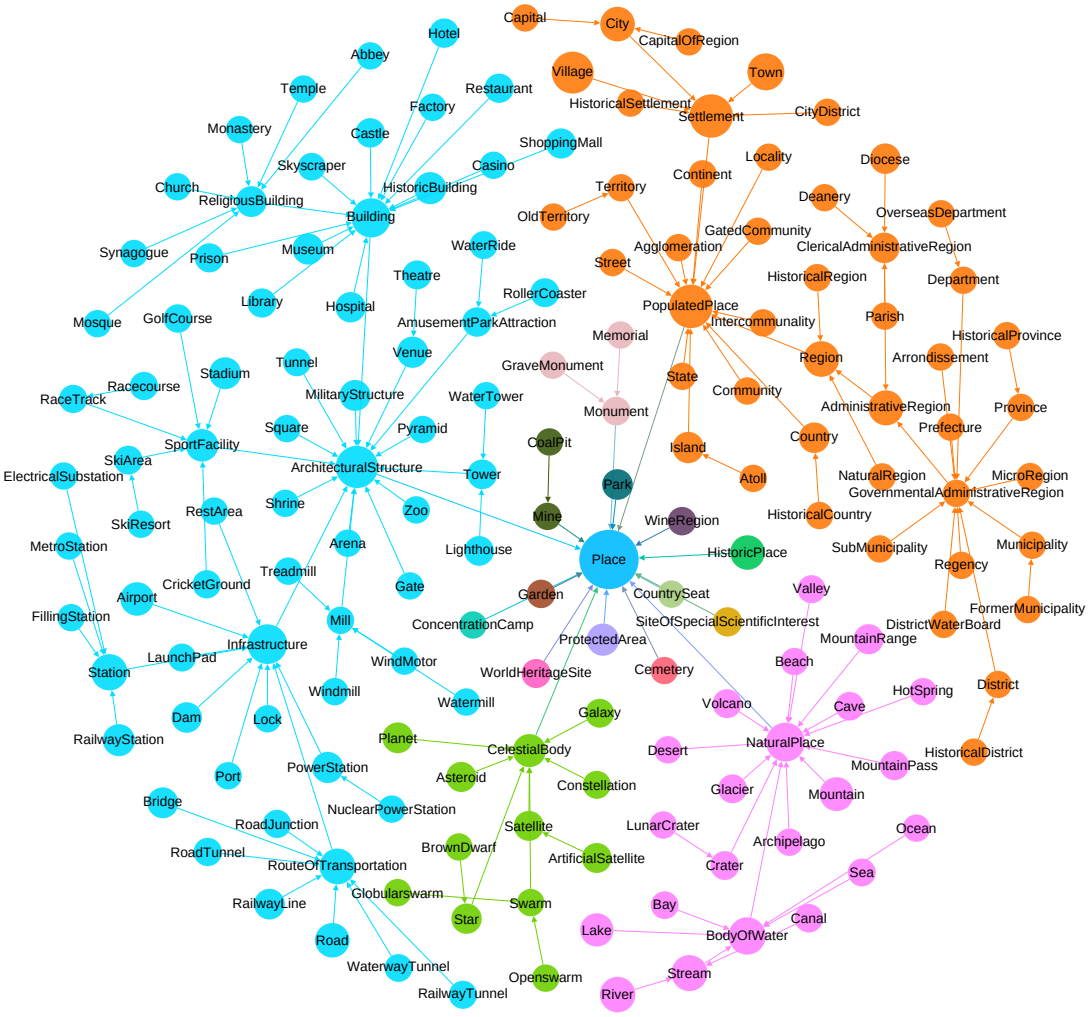

Figure 4.3: The network of place ontology in DBpedia

Town are similar and they have similar attributes (properties) such as populationOf and totalAreaOf. But *City* and *Mountain* are very different from each other because a mountain can have elevation whereas a city usually does not. Based on this train of thought, we build a feature set that shows the relative frequency of each property in each class. We use the following query to extract our feature set from DBpedia Linked Dataset. The result of this query is normalized by the total number of instances of each class to obtain the relative frequency.

```
SELECT ?p (COUNT(?p) AS ?cnt)
```

```
FROM <http://dbpedia.org>
WHERE {
        ?s a :SomeClass .
        ?s ?p ?o .
        FILTERS(contains(str(?p),'dbpedia'))
} GROUP BY ?p
```

In pursuance of comparing the results of our data-driven approach, we also consider different variations of the feature set. We take into account four variables in our feature selection. They are *filler*, *specificity*, *literal* and *uniformity*. All of them are boolean variables. Table 4.1 gives a detailed explanation of each variable. The variable *filler* decides whether we use {property, object type} pairs or property alone to count the frequency. The variable *specificity* takes into consideration the hierarchical structure of object types. The variable *literal* acknowledges the fact that, in RDF, object and literal have different typing schemes, namely object type and data type. The object type is defined by rdf:type whereas the data type is defined by an annotation at the end of the literal string. The variable *uniformity* considers the cases in which the literal of the same property has different data types because the original Wikipedia page does not define a uniform data type for each infobox entry. For example, a city may have a property population, but the literal value for this property may be of type integer or double or even string. In a nutshell, *filler* and *literal* decide whether we incorporate object type and literal type into our feature extraction; *specificity* and *uniformity* deals with the granularity and accuracy of object type and literal type respectively.

Considering all the variables listed here, we have seven feature sets. This whole process of feature selection can be viewed as a decision making process, visualized in a decision tree shown in Figure 4.4. We make a boolean decision on one of the four variables

Table 4.1: Definition for four variables

	True	False
Filler	Include object types	Do not include object types
Specificity	Include only the most specific object types	Include all object types
Literal	Include literal types	Do not include literal types
Uniformity	Unify literal types	Do not unify literal types

described above at each internal node. Each internal node branches into two sub-nodes which are the outcomes of the two decisions based on each of the four variables in our feature selection process. The seven leaf nodes are the final outcomes of the decision tree, which are also the seven feature sets.

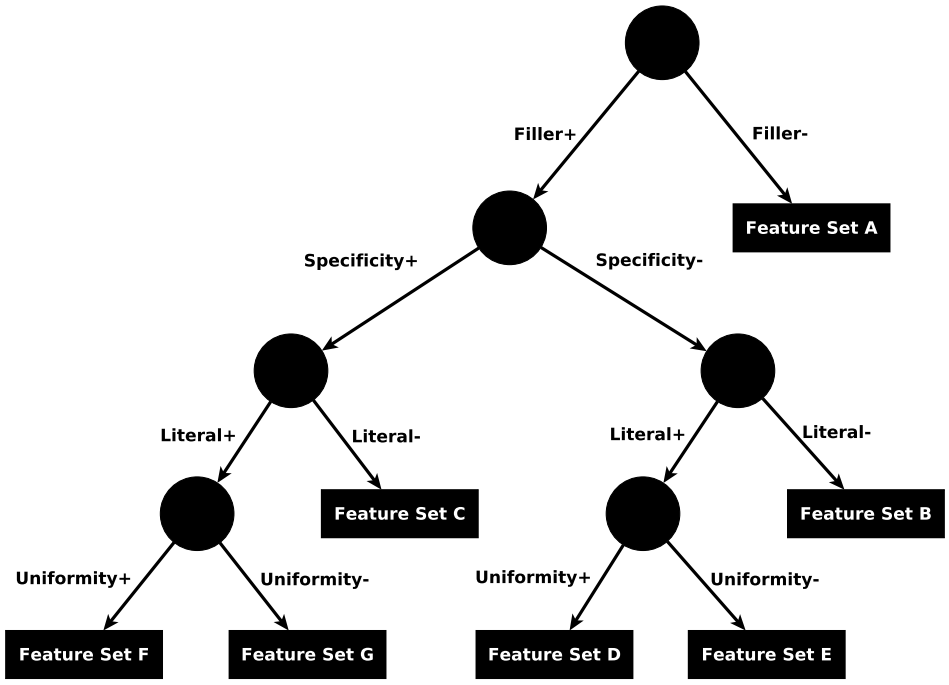

Figure 4.4: The tree structure of feature sets

After analyzing the class vector data, we find that many class vectors have zero frequency for all features, due to the fact that many classes have no populating instances. It makes no sense to include these empty classes in the algorithm since they don't have any data associated with them. In addition, due to the nature of the hierarchical clustering

algorithms we are going to use later, we removed non-leaf classes, otherwise superclasses will be in the same hierarchy with their subclasses. Table 4.2 shows the dimensions of each feature set. Table 4.3 shows the dimensions of each feature set after applying dimensionality reduction algorithms. For example, Feature Set E originally has 68 classes and each class has 47532 features, so the original dimension for Feature Set E is 68 by 47532. After dimensionality reduction, there are only 63 features, so the dimension is 68 by 63. We will use these results as the inputs for clustering algorithms.

Table 4.2: Feature set statistics

	# of Classes × # of Features
Feature Set A	68 × 15918
Feature Set B	68 × 25009
Feature Set C	68 × 13802
Feature Set D	68 × 40284
Feature Set E	68 × 47532
Feature Set F	68 × 29077
Feature Set G	68 × 36328

Table 4.3: Dimensions after MDS

	# of Classes × # of Features
Feature Set A	68 × 62
Feature Set B	68 × 46
Feature Set C	68 × 37
Feature Set D	68 × 64
Feature Set E	68 × 63
Feature Set F	68 × 63
Feature Set G	68 × 62

4.3 Experiment Procedure

In order to apply clustering algorithms, we need to transform the original feature sets into distance matrices. Figure 4.5 - 4.11 show the distance matrices for some of the classes from Feature Set A to Feature Set G respectively.

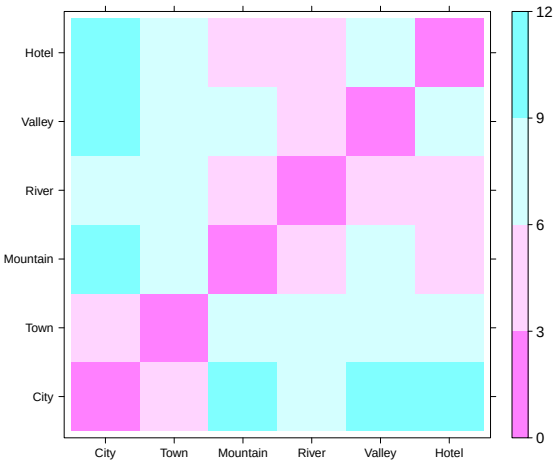

Figure 4.5: Distance matrix from Feature Set A

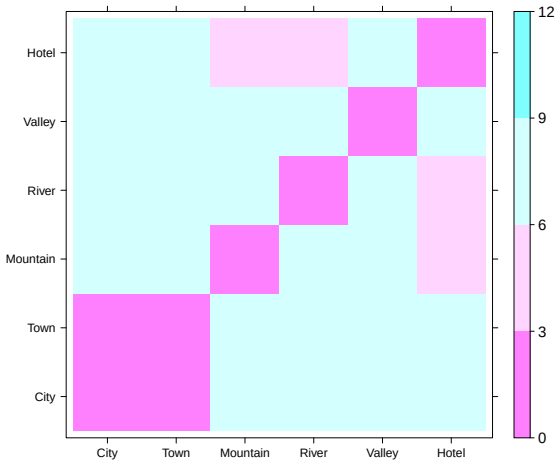

Figure 4.6: Distance matrix from Feature Set B

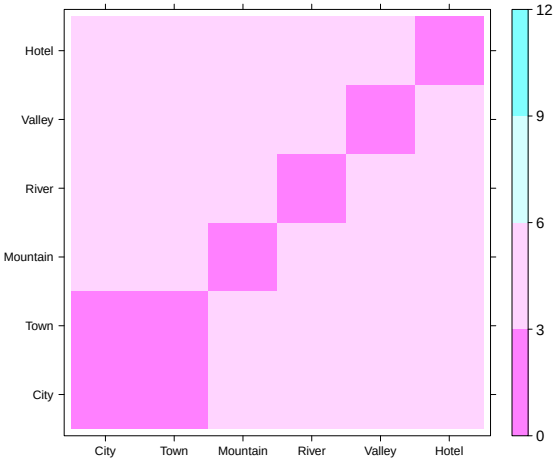

Figure 4.7: Distance matrix from Feature Set C

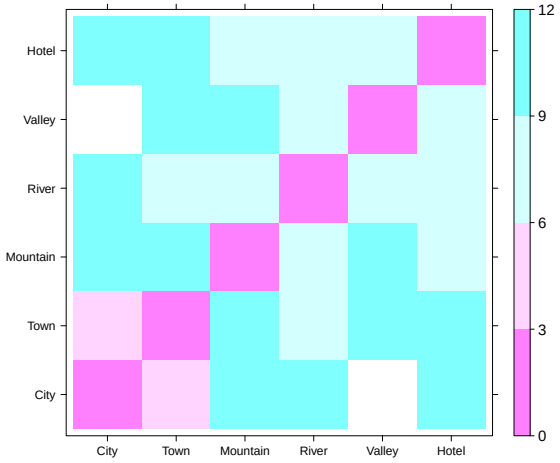

Figure 4.8: Distance matrix from Feature Set D

From the matrices we can tell that natural features are quite distinct from human settlements such as city and town regardless of the types of features choosen. Feature Set C makes the classes more homogeneous in terms of the distance in feature space while Feature Set D makes them more heterogeneous. Although the distance matrices has already revealed some of the information within the dataset and conform with human interpretation of these classes to a certain degree, it's hard to derive any fruitful result from the matrices alone since a 68 by 68 matrix visualization is far from ideal for human

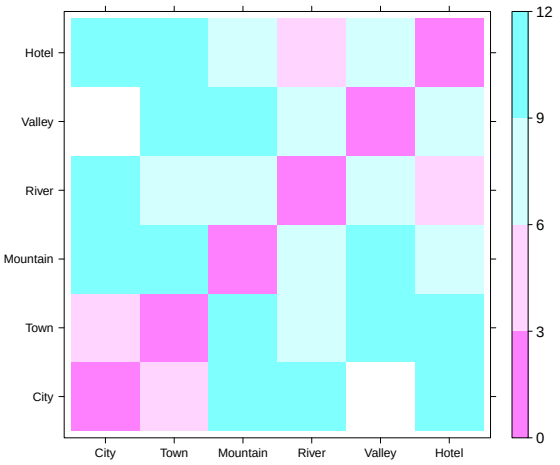

Figure 4.9: Distance matrix from Feature Set E

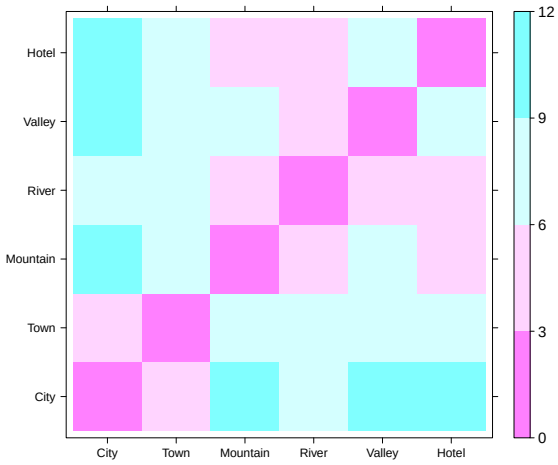

Figure 4.10: Distance matrix from Feature Set F

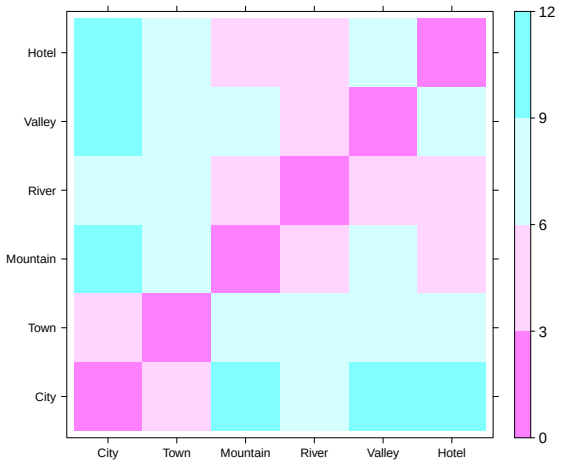

Figure 4.11: Distance matrix from Feature Set G

analysis.

The results of the hierarchical clustering of the 68 classes using Euclidean distance and Ward's method are shown from Figure 4.12 to Figure 4.18.

At this moment, we have constructed seven different class hierarchies from seven different feature sets using data-driven approaches. In hope of further investigate the extent to which our proposed data-driven approaches can correlate with the original labor-intensive man made ontology, we need to apply our validation procedure. Our null

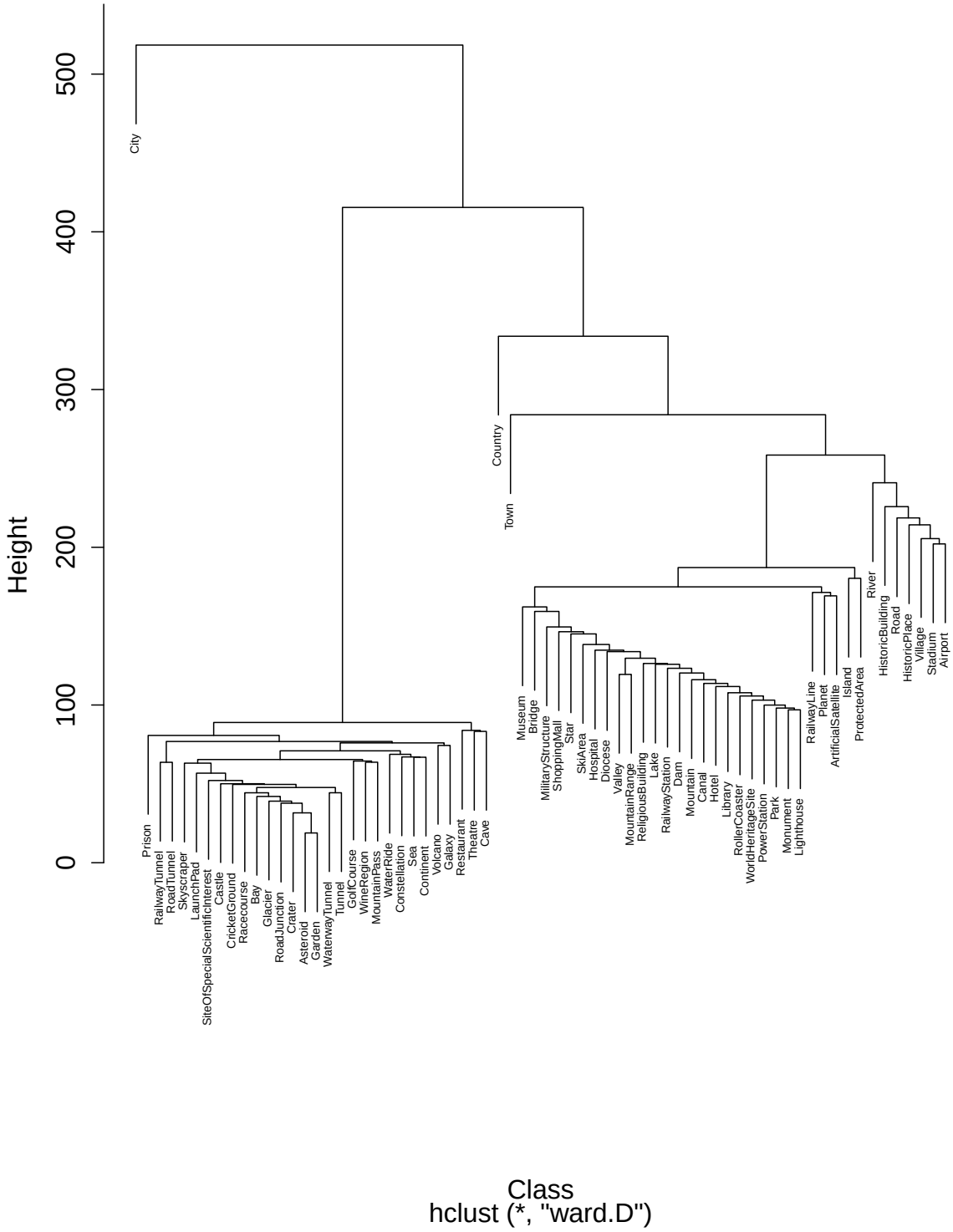

Figure 4.12: Dendrogram for Feature Set A

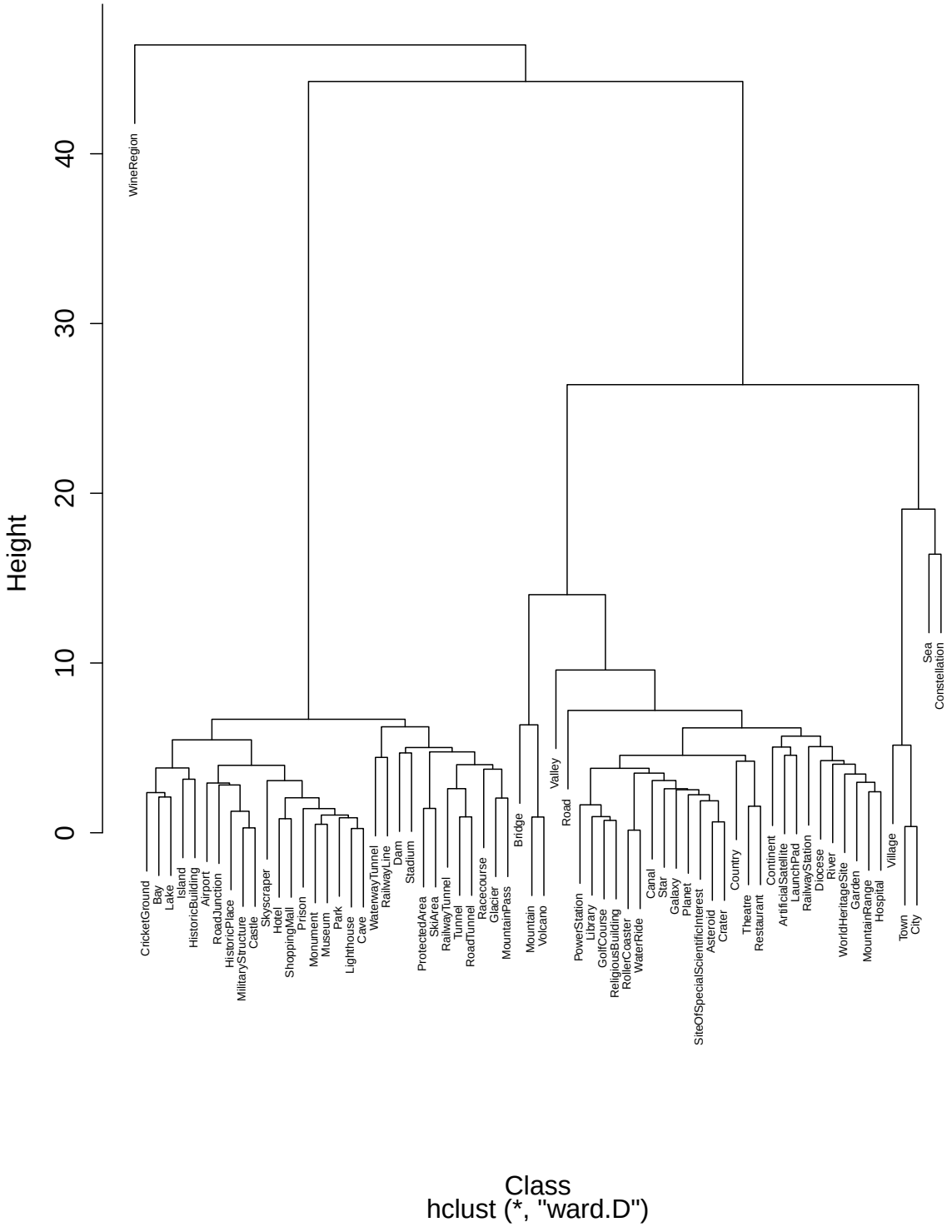

Figure 4.13: Dendrogram for Feature Set B

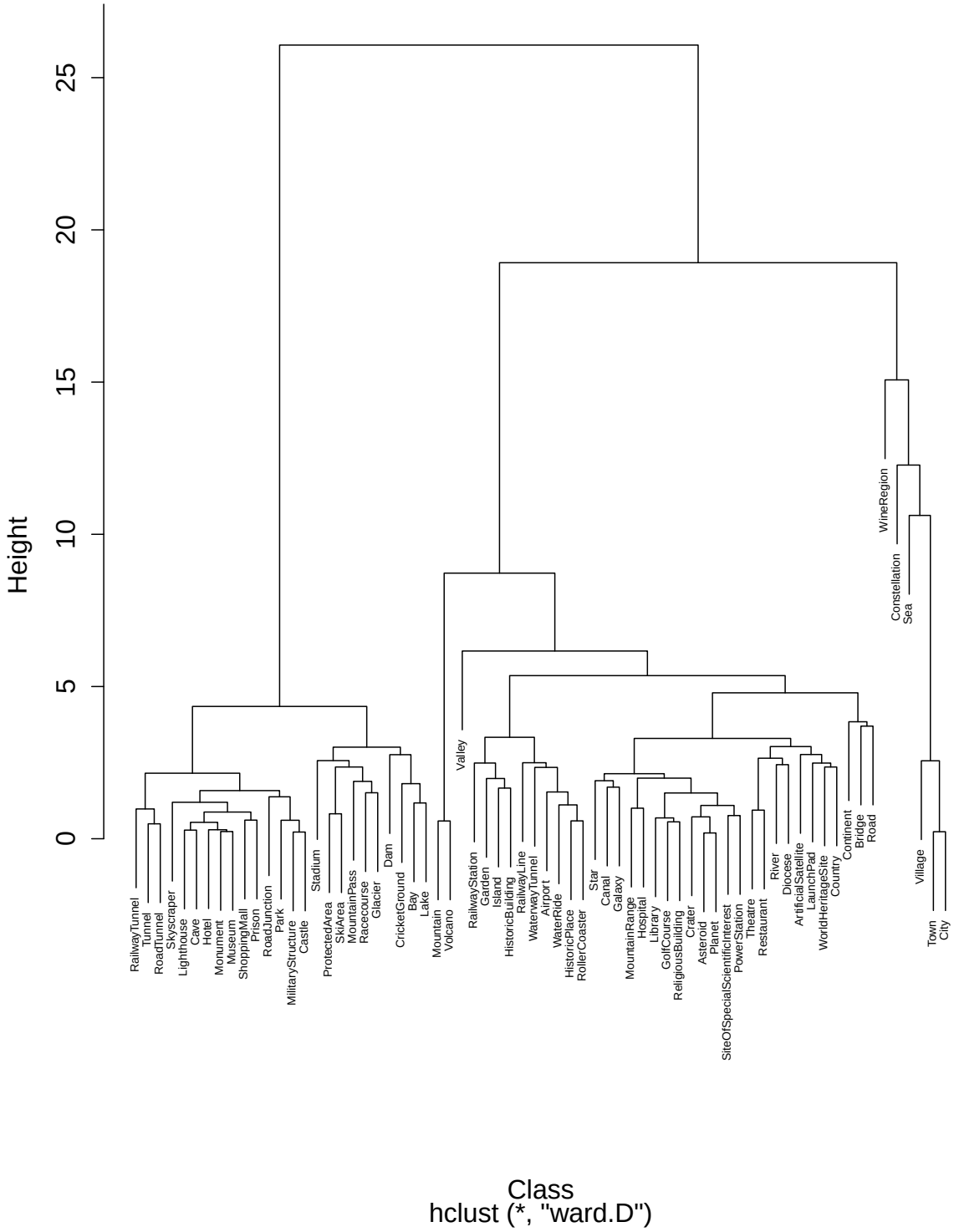

Figure 4.14: Dendrogram for Feature Set C

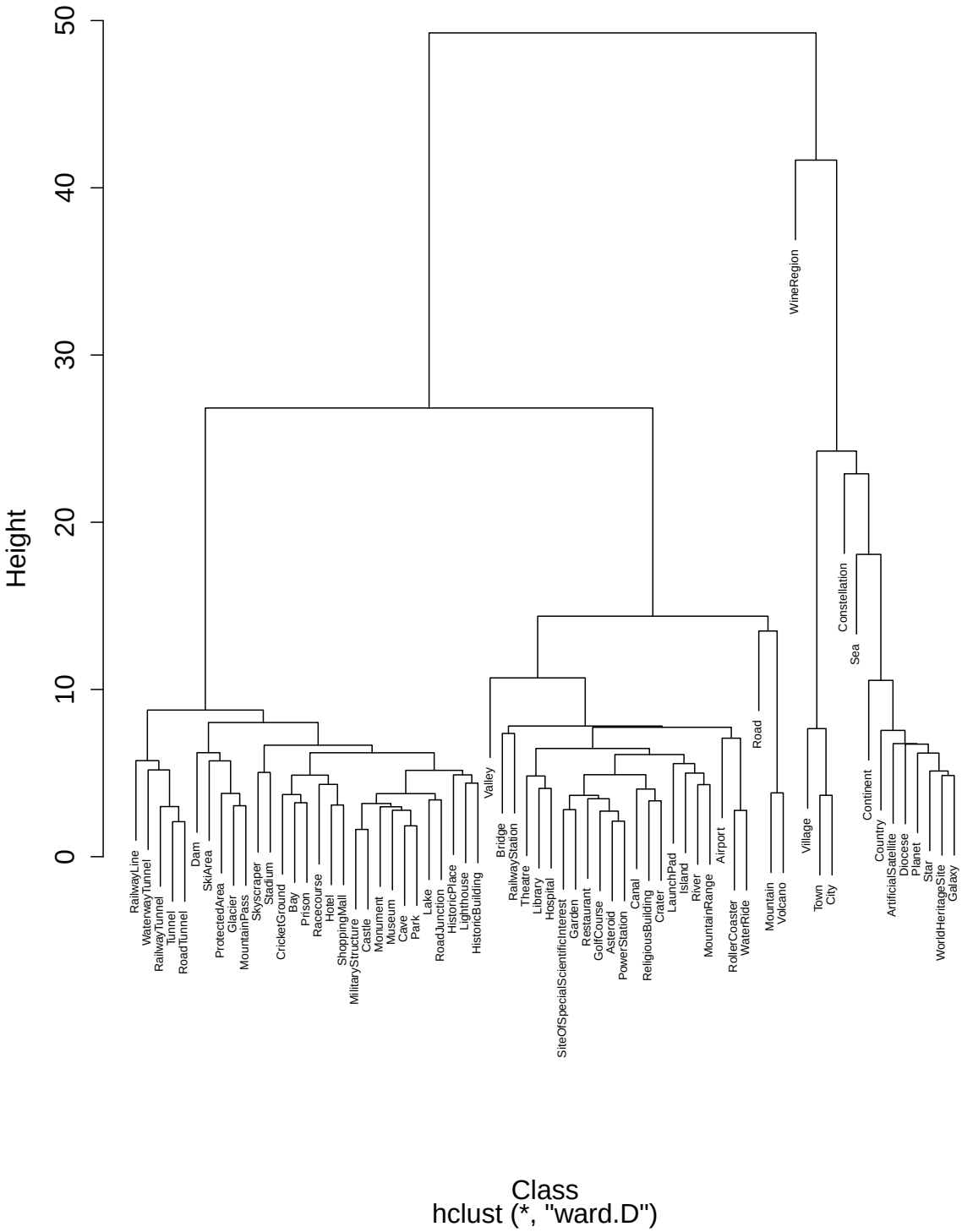

Figure 4.15: Dendrogram for Feature Set D

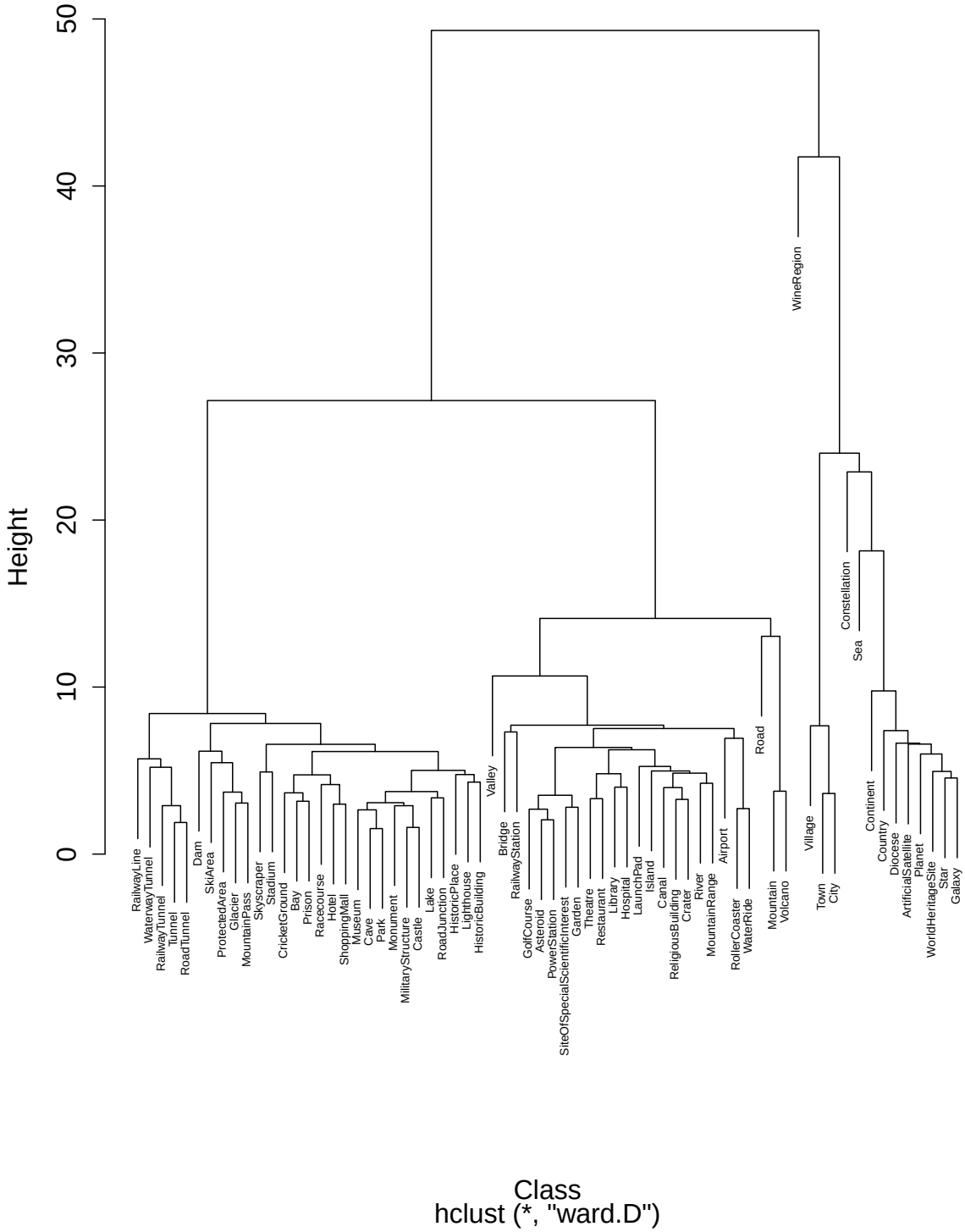

Figure 4.16: Dendrogram for Feature Set E

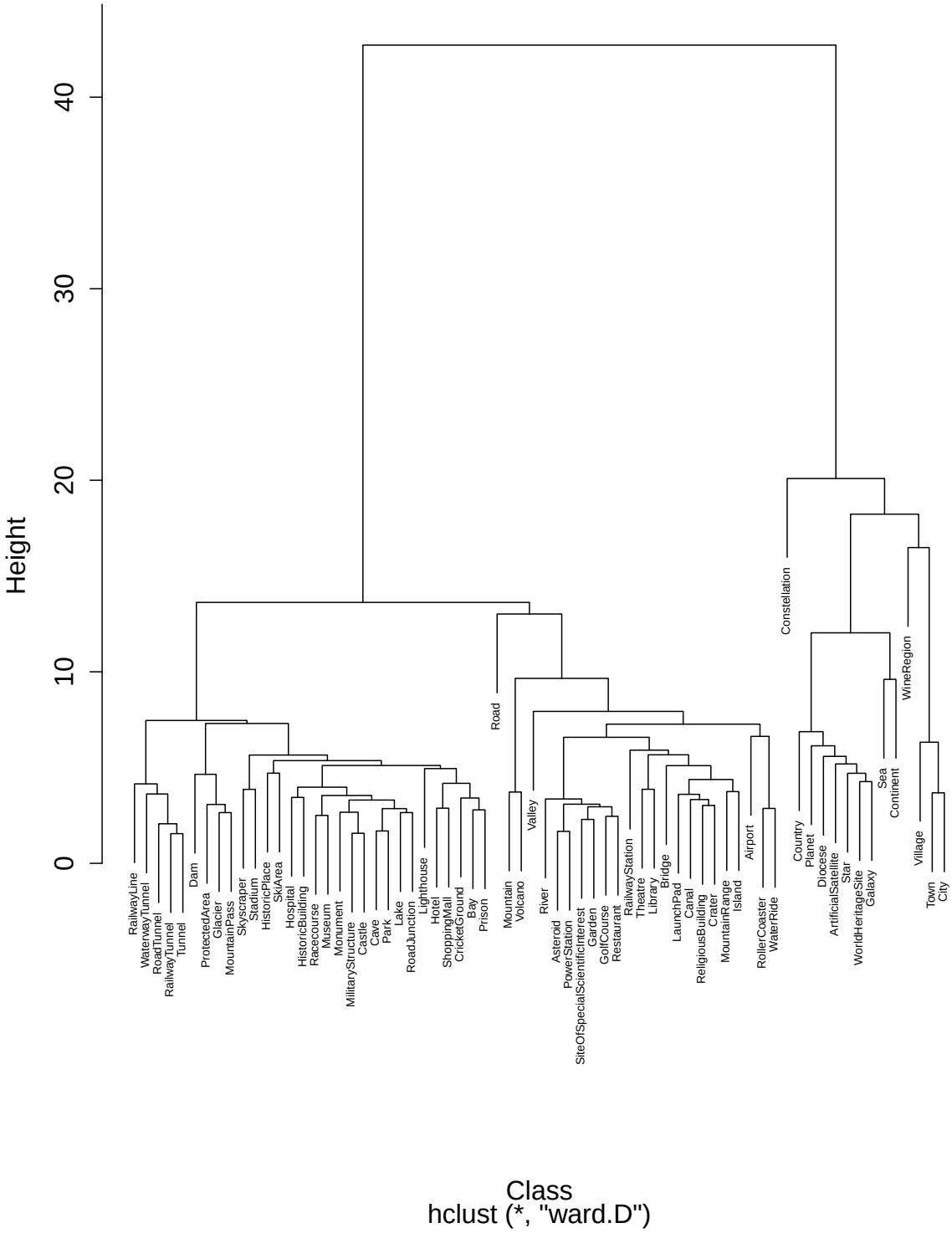

Figure 4.17: Dendrogram for Feature Set F

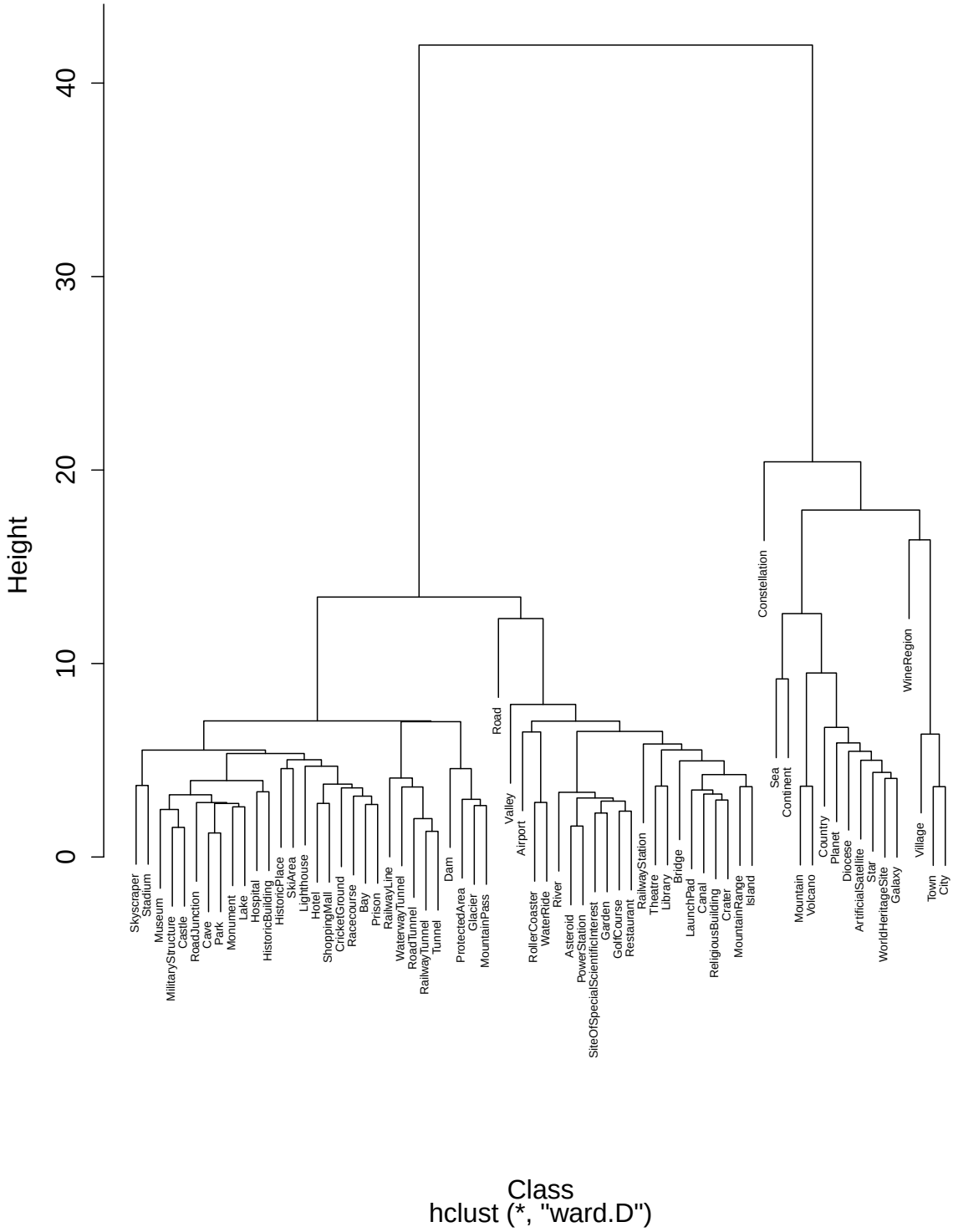

Figure 4.18: Dendrogram for Feature Set G

hypothesis is that there is no correlation between the derived hierarchy and the original hierarchy.

Figure 4.19 shows the p-value and correlation coefficient obtained after running the Mantel test between the original ontology hierarchy and the hierarchies derived from the seven different feature sets. Based upon the p-value resutls, we can reject the null hypothesis for the last six feature sets. There are positive relationships between the derived hierarchical structures using our approaches and the original one. Moreover, the result from Feature Set E has a slightly higher correlation compared with others. Thus, in the rest of this chapter, we focus on Feature Set E.

These hierarchies are very similar to the original hierarchy in the geo-ontology, but, instead of grouping them based on domain knowledge, classes are grouped together based on their properties. Classes clustered together have similar properties in the Linked Dataset. Visually comparing these hierarchies with the original one can give us some information about the difference between the ontology and the Linked Dataset, but we still need some formalized metrics to quantify and measure the difference. In this case, it is necessary for us to find a method to compare different hierarchical clusterings. Previous work [57, 58] has been focusing on defining simple statistics for comparing hierarchical clusterings. However, a simple statistic or index is not useful in terms of measuring the mismatch between individual classes in the geo-ontology. In order to achieve this, we need to preserve the hierarchy and derive useful information for individual classes from different hierarchal structures, instead of converting the hierarchical difference into less informative statistics directly. We transform the hierarchies into similarity matrices and compare the difference between similarity matrices. In this way, we are able to compare individual classes and their differences.

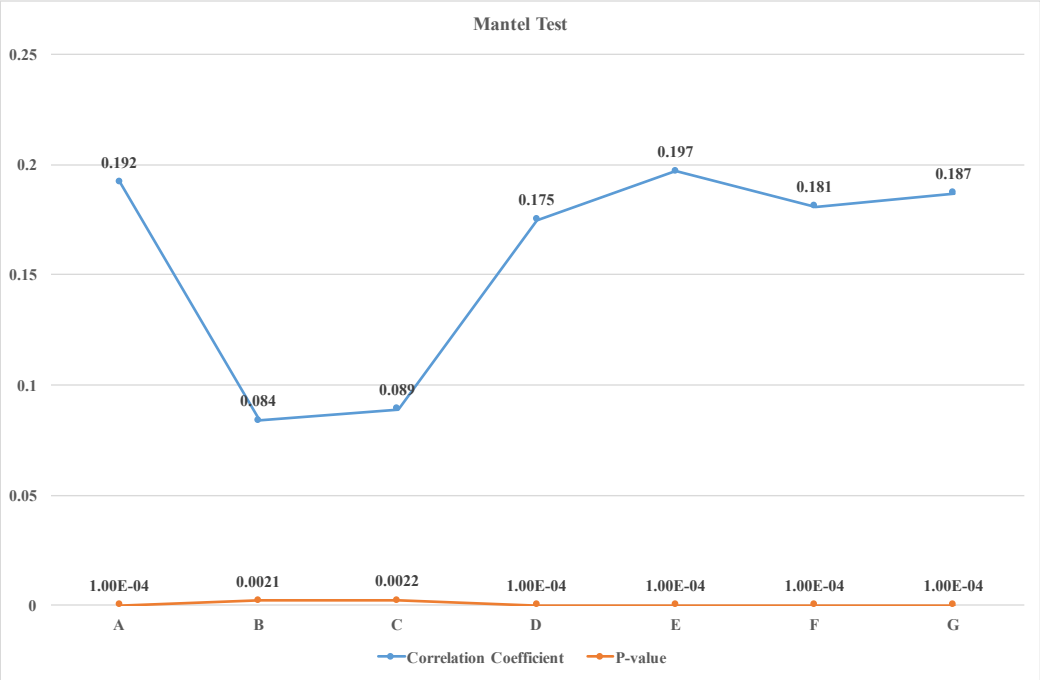

Figure 4.19: Mantel test results for seven feature sets

4.4 Case Studies

In this section, we demonstrate the broad applicability of our data-driven framework and the *discrepancy index* by looking at three examples. The first two examples (Section 4.4.1 and Section 4.4.2) illustrate the ontological modeling issue while the last example illustrates the Linked Data quality issue (Section 4.4.3). We will show how ontology engineers could identify both issues with the help of our methods.

4.4.1 Are *Canals Natural Places?*

Previously, we discovered that *Canal* should not be classified as the subclass of *NaturalPlace* based on its definition. In order to further verify it, we can check if DBpedia Linked Dataset supports our observation. Browsing through the DBpedia page of Panama Canal, one of the seven wonders of modern world listed by the American Society

of Civil Engineers, we found properties such as dbo:principalEngineer, dbp:dateUse and dbp:company (see Figure 4.20). It is really unreasonable for an instance of *NaturalPlace* to have principal engineers, to have the date of first use, and to be originally owned by a company.

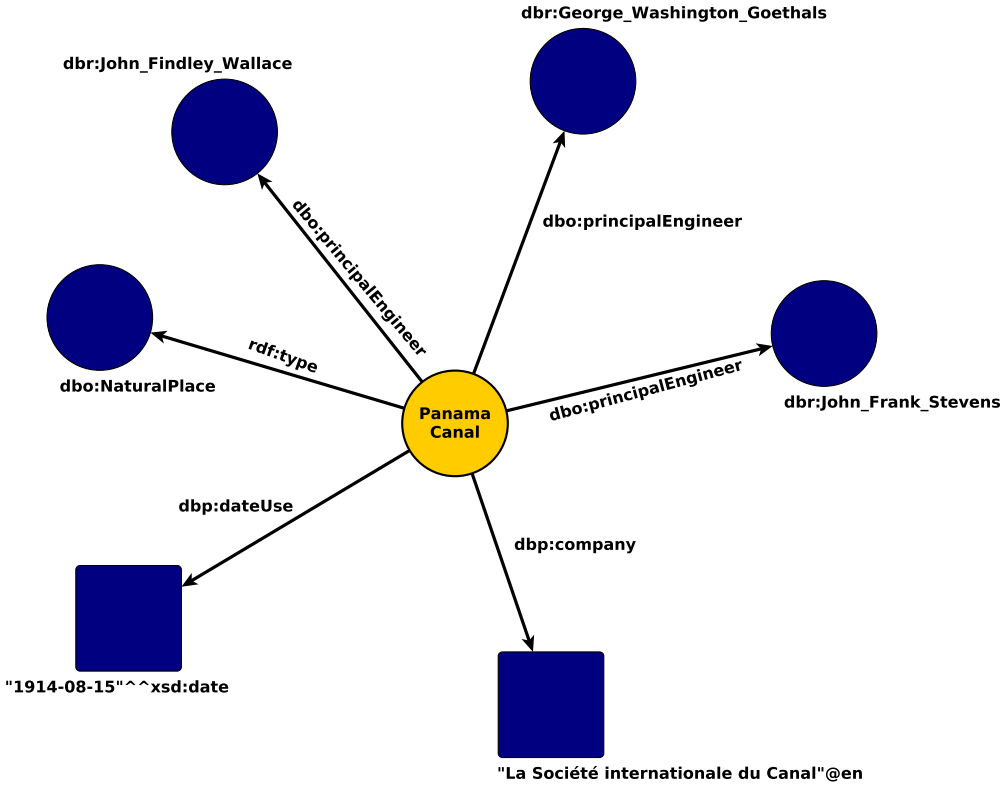

Figure 4.20: Some properties of the instance Panama Canal in DBpedia

It shows that we do see a mismatch between the Linked Data and the geo-ontology in DBpedia by manual inspection. Now let us see how our methods can detect such mismatch and quantify it in order to help geo-ontology engineers in identifying potential ontological modeling issues. We use *Canal* to demonstrate it.

We approach this by comparing the semantic similarity between *Canal* and its sibling classes, in this case, *River*. We would assume that since *City* and *River* are sibling classes, their are semantically very similar to each other.

We first plot the bar graphs for *Canal* and *River*. The bar graph uses information from the feature space after dimensionality reduction. The reason we use the transformed feature space is two-fold. First, the original feature space contains numerous features which are hard to visualize and plot in a graph. Moreover, the original feature space is very sparse, making it difficult to analyze the results in a plotted graph. Second, some of the features are dependent on each other, making some of the information redundant and unreliable. The transformed feature space for Feature Set E only contains 63 features and all of the features are independent from each other.

Figure 4.21 and Figure 4.22 show the bar graphs of these two classes. We call these bar graphs the signatures of these two classes. The x axis represents the features for each class in the same order while the y axis is the value for each feature. From the signatures, we can easily tell that *Canal* has a distinct signature from *River*. However, the signatures are unable to quantify the difference between these classes. Moreover, it is also impossible to obtain the same kind of graphs using the original hierarchy in the ontology. Thus, these signatures can't inform us the direction of ontology refinement and the extent to which we need to modify the ontology.

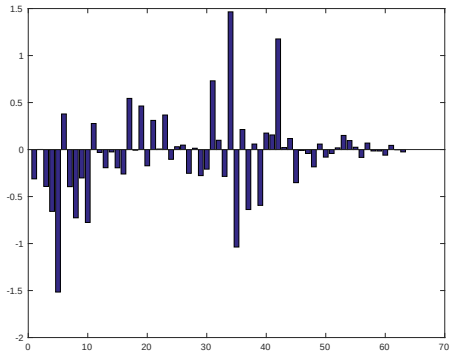

Figure 4.21: Canal

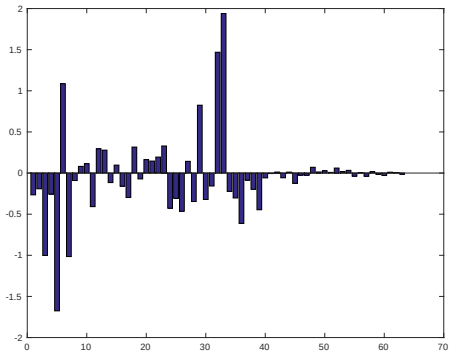

Figure 4.22: River

In this case, we can use the proposed Discrepancy Index to help us. After looking up the similarity matrices for the original hierarchy and the derived hierarchy, we find that

the similarity between *Canal* and *River* is 0.84 and 0.23 respectively. The Discrepancy Index for (*Canal*, *River*) is 0.61. This index implies that *Canal* is less similar to *River* in the derived hierarchy and leads to further investigation, which in this case is the ontological modeling issue. This result corresponds to our observation in the DBpedia geo-ontology.

4.4.2 *City*, *Town* or *Village*

Now let's look at a more interesting case. In DBpedia place ontology, *City*, *Town* and *Village* are classified as sibling classes, meaning they are very similar to each other. The signatures for *City*, *Town* and *Village* in Figure 4.23, Figure 4.24 and Figure 4.25 show a different picture. To further investigate this, we use the Discrepancy Index proposed in this research.

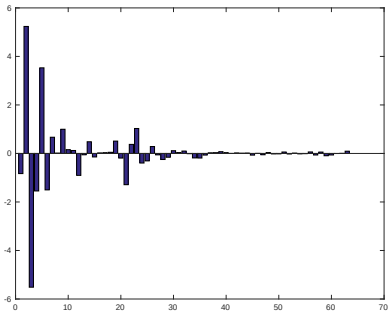

Figure 4.23: City

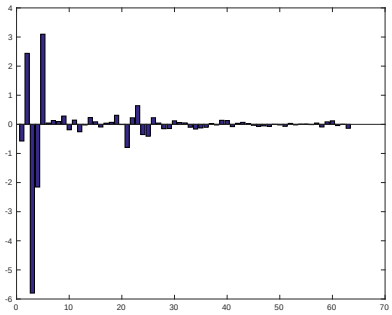

Figure 4.24: Town

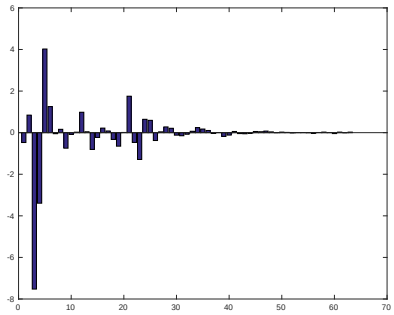

Figure 4.25: Village

Table 4.4 shows the pairwise semantic similarities for *City*, *Town* and *Village* from the original ontology hierarchy. They are all 0.75, which implies that *City*, *Town* and *Village* are equally similar to each other in the original geo-ontology. Table 4.5 shows the pairwise semantic similarities for *City*, *Town* and *Village* from the derived ontology hierarchy. The similarity between *City* and *Town* drops from 0.75 to 0.62. The similarity between *City* and *Village* and the similarity between *Town* and *Village* both drop from 0.75 to 0.49.

Table 4.4: Pairwise similarities from the original ontology hierarchy

	City	**Town**	**Village**
City	1	0.75	0.75
Town	0.75	1	0.75
Village	0.75	0.75	1

Table 4.5: Pairwise similarities from the derived ontology hierarchy

	City	**Town**	**Village**
City	1	0.62	0.49
Town	0.62	1	0.49
Village	0.49	0.49	1

Table 4.6: Discrepancy Indices for *City*, *Town* and *Village*

	City	**Town**	**Village**
City	0	0.13	0.26
Town	0.13	0	0.26
Village	0.26	0.26	0

Table 4.6 shows the Discrepancy Indices for *City*, *Town* and *Village*. It shows that *City* and *Town* are less similar in the derived geo-ontology hierarchy. (*Town*, *Village*) and (*City*, *Village*) are also less similar and to a greater degree. This result reveals some ontological modeling issues and can lead to some interesting discussion about *City*, *Town* and *Village*.

When it comes to the difference among *City*, *Town* and *Village*, there never is an unanimous conclusion. Different countries and regions define them differently based on local law or tradition. Even within the United States, they are treated differently in different states. For instance, in California and Maryland, *City* and *Town* are legally the same, whereas, in Utah and Alabama, *City* and *Town* are used differently based upon population. In this sense, they are different because of the population size. The subtlety in the mismatch between the ontology and the Linked Dataset also signifies that, to a certain degree, *City* and *Town* are more similar with each other than with *Village*.

4.4.3 *Hotel* in DBpedia

In the original ontology hierarchy, *Hotel* is classified as the sibling class of *Restaurant*, *Museum* and *ShoppingMall* under the superclass *Building*. This is a reasonable classification and corresponds to our perception of these concepts.

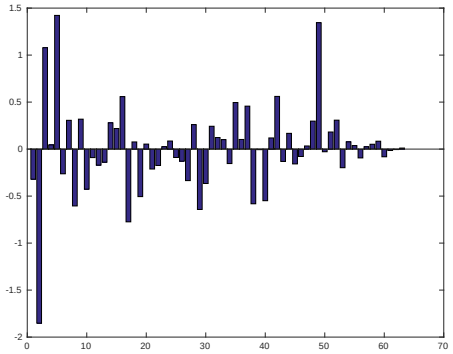

Figure 4.26: Hotel

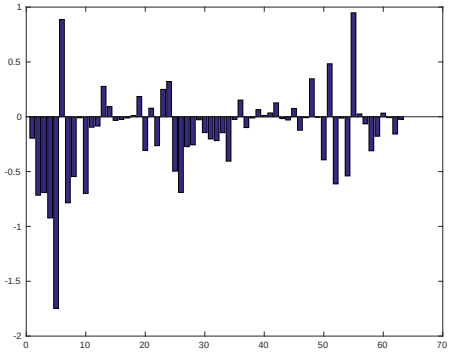

Figure 4.27: Restaurant

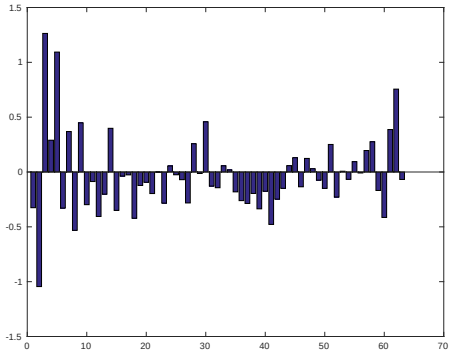

Figure 4.28: Museum

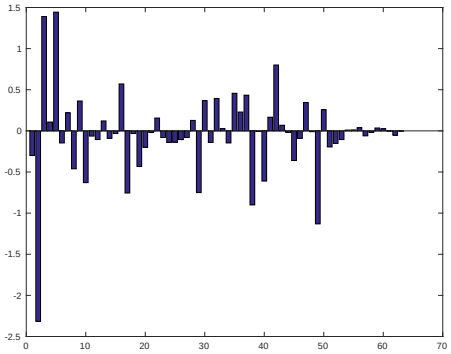

Figure 4.29: Shopping Mall

Table 4.7: Similarity in original and derived hierarchy

	Restaurant	**Museum**	**ShoppingMall**
Original	0.58	0.58	0.58
Derived	0.14	0.29	0.73

The semantic similarities in the original ontology hierarchy and the derived hierarchy with respect to *Hotel* are shown in Table 4.7. It shows that the similarity between *Hotel*

and *Restaurant* drops from 0.58 to 0.14, the similarity between *Hotel* and *Museum* drops from 0.58 to 0.29, and the similarity between *Hotel* and *ShoppingMall* increases from 0.58 to 0.73. The Discrepancy Index for (*Hotel*, *Restaurant*) is 0.44. The Discrepancy Index for (*Hotel*, *Museum*) is 0.29. The Discrepancy Index for (*Hotel*, *ShoppingMall*) is -0.15. These discrepancies, though insignificant, draw our attention and demand some investigation into the cause. By querying these classes we are able to find that these discrepancies may be caused by data bias in DBpedia Linked Dataset because *Hotel* surprisingly only has 1237 instances.

Chapter 5

Conclusions and Future Work

This research provides a workflow of an ontology engineering assistance system based upon the fact that ontology engineering is a systematic and dynamic cycle of multiple processes. This chapter synthesize the experiment results and conclude on the contributions and rationale behind this research. Directions for future areas of research are also presented in this chapter.

5.1 Conclusions

The current ontology construction procedure depends heavily on the knowledge of domain experts. The potential pitfall to this routine is that the resulting ontology maybe biased and sometimes is not representative of our perception of the world. Furthermore, this practice is tedious and time-consuming. DBpedia initiated a community-based endeavor to distribute the labor-intensive ontology construction task to general users. This initiative is helpful in terms of mitigating the current difficulty in ontology engineering by crowd-sourcing the effort of modeling our world. However, due to the complexity and subtlety of ontology engineering, general users are frequently intimidated by the

formidable jargons and rules in this task. Furthermore, this distributed way of ontology engineering does not provide solutions to the inherent modeling bias.

In addition, Linked Data quality issue is also a significant factor that is usually neglected by geo-ontology engineers while creating the geo-ontology. We use an example from DBpedia that takes advantage of its ontology to cleanse the Linked Dataset and provide users with cleaner and better structured data to show that ontology itself can indeed help identify and even deal with Linked Data quality issue.

The state of the art of geo-ontology engineering separates Linked Data from the process. Some recent research incorporate the corresponding Linked Dataset to enrich the top-down ontology. We take a step further and argue that the top-down ontology can also help improve the corresponding Linked Dataset. Considering the two common issues above, namely the ontological modeling issue and Linked Data quality issue, we close the gap in geo-ontology engineering by proposing a data-driven framework that can assist geo-ontology engineers.

By closely examining the Linked Dataset corresponding to the ontology, we build several feature spaces that are representative of the properties and instances in the classes based on four variables, namely *filler*, *specificity*, *literal* and *uniformity*. We then use thess feature spaces to derive the data-driven hierarchical structures. Instead of comparing the hierarchies directly using simple statistics, we transform the hierarchies into similarity matrices. Each similarity matrix contains pairwise similarity between different classes. After verifying the correlations between the data-driven hierarchies and the original hierarchy in the ontology using Mantel test, we select the most correlated one. We use Discrepancy Indices to quantify the discrepancy between the data and the original ontology. These indices can give guidance on correcting the potential ontological modeling issues (such as misclassification) and/or spotting the Linked Data quality issue (such as missing data). We use three examples to showcase the usefulness of our frame-

work. Two of the examples are used to show how this framework can be implemented to detect potential ontological modeling issues. One example is used to show that this framework can also help identify potential Linked Data quality issue. The experiment result is consistent with our initial observation of DBpedia place ontology and Linked Dataset.

Since ontology engineering is a dynamic process, this assistance framework can be implemented throughout the life cycle of ontology engineering, making it an invaluable tool. The performance of the proposed workflow depends on the quality of the training dataset. So data cleaning and noise reduction are important. It is also important to point out that, although the workflow described in this research is applied using DBpedia geo-ontology and its Linked Dataset, it is also viable when applied to other ontologies and Linked Dataset, such as GeoNames and LinkedGeoData.

In this work, we have explored different combinations of feature extraction strategies. The Mantel test results show that the correlation coefficient does not increase monotonically as we move from Feature Set A to Feature Set B. It decreases first and then increases. This finding is different from our expectations in that we intuitively think that as we move from simple to complex feature combinations, the correlations will increase monotonically. The reason for the result perhaps lies in the way the geo-ontology is constructed in the first place: rather than adopt an increasingly sophisticated definition for each class, domain experts may have made efforts in striking the right balance between complexity and simplicity of the selected concepts and attributes. We may be able to render hierarchies that have even higher correlation with the original geo-ontology hierarchy if we extract features other than properties.

5.2 Future Work

This research can be extended in several aspects. First of all, our experiment so far focuses on a particular ontology and dataset. With a wide range of availability of Linked Data and ontologies on the Web, we can test our framework using difference data sources to further justify the usefulness of our approach. The challenge of applying this framework to other geo-ontologies and geospatial Linked Dataset is that we need to transform them into a standard format that is ready to be used in the framework. For example, Schema.org ontology does not provide a standard OWL file for the ontology and it is generated from a large number of webpages and resource that are hard to be processed in a single database. Moreover, some Linked Dataset is so huge that it poses a burden the triple store and querying them efficiently has become an important task to accomplish. The efficiency of SPARQL and GeoSPARQL queries has been an active research area. Future research will likely benefit from the progress in this area.

Second, the current workflow only helps in detecting the discrepancy between the ontology and the dataset, but it does not provide candidate solutions. For instance, after detecting the misclassification of *Canal*, it will be useful if the ontology engineering assistance system can provide some candidate classes, such as *WaterwayTunnel*, for suggestions. The challenge for providing candidates is that this task is mainly subjective and require a lot of domain knowledge. In order to provide reasonable candidates, the framework will need to be adjusted to allow more comprehensive feature selection criteria.

Third, there is still development on the feature selection step in order to find more suitable feature sets for the data mining algorithms used in this work. For example, in addition to the four variables considered in this work, the hierarchy of properties can also be taken into consideration, e.g. *dbo:isPartOfRoute* is a subproperty of *dbo:isPartOf*.

Furthermore, instead of focusing exclusively on the attributes of place classes, future research could also extract implicit topological relationships among different place entities and classes. In this way, we can enrich our features and these new features can help us detect more information regarding the Linkeda Dataset and the corresponding ontology. The challenge of adopting a richer set of features also lies in improving the efficiency of SPARQL queries.

Fourth, the current workflow in this research cannot differentiate whether the discrepancy is due to the ontology modeling bias or the bias in the Linked Dataset. For the simple reason that both issues coexist in current Linked Data and ontology, differentiating them based purely on data-driven approaches requires certain ground truth. Such ground truth can be surveys on people's perception of the classes and relationships in question. The challenge is that, since these perceptions are largely subjective, there might not be a meaningful agreement on certain issues. Future work can further investigate in this direction to make the assistance system more intelligent.

Bibliography

[1] R. V. Guha, *Light at the end of the tunnel*, *Proceedings of ISWC* (2013).

[2] A. Singhal, "Introducing the knowledge graph: things, not strings." `https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html`, 2012. 2016-02-01.

[3] R. Guha, D. Brickley, and S. Macbeth, *Schema. org: evolution of structured data on the web*, *Communications of the ACM* **59** (2016), no. 2 44–51.

[4] W3C, "Linked data." `https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html`. 2016-02-01.

[5] Schema.org, "About schema.org." `https://schema.org/docs/about.html`. 2016-02-01.

[6] M. F. Goodchild, *Formalizing place in geographic information systems*, in *Communities, Neighborhoods, and Health*, pp. 21–33. Springer, 2011.

[7] K. Janowicz, S. Scheider, T. Pehle, and G. Hart, *Geospatial semantics and linked spatiotemporal data-past, present, and future*, *Semantic Web* **3** (2012), no. 4 321–332.

[8] Y. Hu and K. Janowicz, *Enriching top-down geo-ontologies using bottom-up knowledge mined from linked data*, in *H. Onsrud & W. Kuhn (Eds), Advancing Geographic Information Science: The Past and Next Twenty Years*, pp. 183–198, GSDI Association Press, 2016.

[9] K. Janowicz, *Observation-driven geo-ontology engineering*, *Transactions in GIS* **16** (2012), no. 3 351–374.

[10] M. Knuth, D. Kontokostas, and H. Sack, *Linked data quality: Identifying and tackling the key challenges.*, in *LDQ@ SEMANTICS*, 2014.

[11] J. J. Jiang and D. W. Conrath, *Semantic similarity based on corpus statistics and lexical taxonomy*, *arXiv preprint cmp-lg/9709008* (1997).

[12] A. Maedche, *Ontology learning for the semantic web*, vol. 665. Springer Science & Business Media, 2012.

[13] M. Hazman, S. R. El-Beltagy, and A. Rafea, *A survey of ontology learning approaches*, *database* **7** (2011) 6.

[14] A. Gómez-Pérez and D. Manzano-Macho, *A survey of ontology learning methods and techniques*, 2003.

[15] M. Shamsfard and A. A. Barforoush, *The state of the art in ontology learning: a framework for comparison*, *The Knowledge Engineering Review* **18** (2003), no. 04 293–316.

[16] J. Lehmann and J. Voelker, *An introduction to ontology learning*, *Perspectives on ontology learning. AKA/IOS Press, Heidelberg* (2014) 9–16.

[17] N. Aussenac-Gilles, B. Biébow, and S. Szulman, *Corpus analysis for conceptual modelling*, in *AussenacGilles, N., Biébow, B. & Szulman, S. (Eds.) Paper presented at the Workshop on Ontologies and Texts - 12th International Conference on Knowledge Engineering and Knowledge Management*, 2000.

[18] P. Cimiano, A. Hotho, and S. Staab, *Learning concept hierarchies from text corpora using formal concept analysis.*, *J. Artif. Intell. Res.(JAIR)* **24** (2005) 305–339.

[19] B. Ganter and R. Wille, *Formal concept analysis: mathematical foundations*. Springer Science & Business Media, 2012.

[20] Z. Harris, *Mathematical structures of language*. Wiley, 1968.

[21] L. Khan and F. Luo, *Ontology construction for information selection*, in *Tools with Artificial Intelligence, 2002.(ICTAI 2002). Proceedings. 14th IEEE International Conference on*, pp. 122–127, IEEE, 2002.

[22] J. Dopazo and J. M. Carazo, *Phylogenetic reconstruction using an unsupervised growing neural network that adopts the topology of a phylogenetic tree*, *Journal of molecular evolution* **44** (1997), no. 2 226–233.

[23] G. A. Miller, *Wordnet: a lexical database for english*, *Communications of the ACM* **38** (1995), no. 11 39–41.

[24] T. Joachims, *A probabilistic analysis of the rocchio algorithm with tfidf for text categorization.*, tech. rep., DTIC Document, 1996.

[25] T. Kohonen, *The self-organizing map*, *Proceedings of the IEEE* **78** (1990), no. 9 1464–1480.

[26] E. M. Voorhees, *Implementing agglomerative hierarchic clustering algorithms for use in document retrieval*, *Information Processing & Management* **22** (1986), no. 6 465–476.

[27] C. Papatheodorou, A. Vassiliou, and B. Simon, *Discovery of ontologies for learning resources using word-based clustering*, in *EdMedia: World Conference on Educational Media and Technology*, vol. 2002, pp. 1523–1528, 2002.

[28] M. Perkowitz and O. Etzioni, *Adaptive web sites: Automatically synthesizing web pages*, in *AAAI/IAAI*, pp. 727–732, 1998.

[29] L. Karoui, M.-A. Aufaure, and N. Bennacer, *Ontology discovery from web pages: Application to tourism*, in *In the Workshop of Knowledge Discovery and Ontologies*, Citeseer, 2004.

[30] R. Zarrad, N. Doggaz, and E. Zagrouba, *Toward a taxonomy of concepts using web documents structure*, in *Proceedings of the 14th International Conference on Information Integration and Web-based Applications & Services*, pp. 147–156, ACM, 2012.

[31] H. Schmid, *Probabilistic part-of-speech tagging using decision trees*, in *Proceedings of the international conference on new methods in language processing*, vol. 12, pp. 44–49, Citeseer, 1994.

[32] P. Johannesson, *A method for transforming relational schemas into conceptual schemas*, in *Data Engineering, 1994. Proceedings. 10th International Conference*, pp. 190–201, IEEE, 1994.

[33] V. Kashyap, *Design and creation of ontologies for environmental information retrieval*, in *Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management*, pp. 1–18, Citeseer, 1999.

[34] A. B. Williams and C. Tsatsoulis, *An instance-based approach for identifying candidate ontology relations within a multi-agent system.*, in *ECAI Workshop on Ontology Learning*, 2000.

[35] H. Hlomani and D. Stacey, *Approaches, methods, metrics, measures, and subjectivity in ontology evaluation: A survey*, *Semantic Web Journal* (2014) 1–5.

[36] C. Brewster, H. Alani, S. Dasmahapatra, and Y. Wilks, *Data driven ontology evaluation*, in *Proceedings International Conference on Language Resources and Evaluation*, 2004.

[37] H. Hlomani and D. A. Stacey, *Contributing evidence to data-driven ontology evaluation-workflow ontologies perspective.*, in *KEOD*, pp. 207–213, 2013.

[38] A. Maedche and S. Staab, *Measuring similarity between ontologies*, in *Knowledge engineering and knowledge management: Ontologies and the semantic web*, pp. 251–263. Springer, 2002.

[39] J. Brank, D. Mladenic, and M. Grobelnik, *Gold standard based ontology evaluation using instance assignment*, in *Workshop on Evaluation of Ontologies for the Web, EON*, Edinburgh, UK, 2006.

[40] K. Supekar, *A peer-review approach for ontology evaluation*, in *8th Int. Protege Conf*, pp. 77–79, Citeseer, 2005.

[41] R. Porzel and R. Malaka, *A task-based approach for ontology evaluation*, in *ECAI Workshop on Ontology Learning and Population, Valencia, Spain*, Citeseer, 2004.

[42] H. Paulheim and J. Fümkranz, *Unsupervised generation of data mining features from linked open data*, in *Proceedings of the 2nd international conference on web intelligence, mining and semantics*, p. 31, ACM, 2012.

[43] N. Mantel, *The detection of disease clustering and a generalized regression approach*, *Cancer research* **27** (1967), no. 2 Part 1 209–220.

[44] R. E. Bellman, *Adaptive control processes: a guided tour*. Princeton university press, 2015.

[45] M. A. Carreira-Perpinan, *Continuous latent variable models for dimensionality reduction and sequential data reconstruction*. PhD thesis, University of Sheffield UK, 2001.

[46] A. Buja, D. F. Swayne, M. L. Littman, N. Dean, H. Hofmann, and L. Chen, *Data visualization with multidimensional scaling*, *Journal of Computational and Graphical Statistics* **17** (2008), no. 2 444–472.

[47] H. F. Kaiser, *The application of electronic computers to factor analysis.*, *Educational and psychological measurement* (1960).

[48] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*, vol. 112. Springer, 2013.

[49] K. Janowicz, M. Raubal, and W. Kuhn, *The semantics of similarity in geographic information retrieval*, *Journal of Spatial Information Science* **2011** (2011), no. 2 29–57.

[50] T. Slimani, *Description and evaluation of semantic similarity measures approaches*, *arXiv preprint arXiv:1310.8059* (2013).

[51] K. Janowicz, P. Maué, M. Wilkes, S. Schade, F. Scherer, M. Braun, S. Dupke, and W. Kuhn, *Similarity as a quality indicator in ontology engineering*, in *Proceedings of the 5th International Conference on Formal Ontology in Information Systems (FOIS)*, vol. 183, pp. 92–105, IOS Press, 2008.

[52] D. Sánchez, M. Batet, and D. Isern, *Ontology-based information content computation*, *Knowledge-Based Systems* **24** (2011), no. 2 297–303.

[53] P. Legendre and M.-J. FORTIN, *Comparison of the mantel test and alternative approaches for detecting complex multivariate relationships in the spatial analysis of genetic data*, *Molecular ecology resources* **10** (2010), no. 5 831–844.

[54] P. N. Mendes, M. Jakob, and C. Bizer, *Dbpedia: A multilingual cross-domain knowledge base.*, in *LREC*, pp. 1813–1817, Citeseer, 2012.

[55] M. Schmachtenberg, C. Bizer, A. Jentzsch, and R. Cyganiak, "Linking open data cloud diagram 2014." `http://lod-cloud.net`, 2014.

[56] L. Stojanovic and B. Motik, *Ontology evolution within ontology editors*, in *Proceedings of the OntoWeb-SIG3 Workshop*, pp. 53–62, 2002.

[57] E. B. Fowlkes and C. L. Mallows, *A method for comparing two hierarchical clusterings*, *Journal of the American statistical association* **78** (1983), no. 383 553–569.

[58] I. Morlini and S. Zani, *An overall index for comparing hierarchical clusterings*, in *Challenges at the interface of data analysis, computer science, and optimization*, pp. 29–36. Springer, 2012.